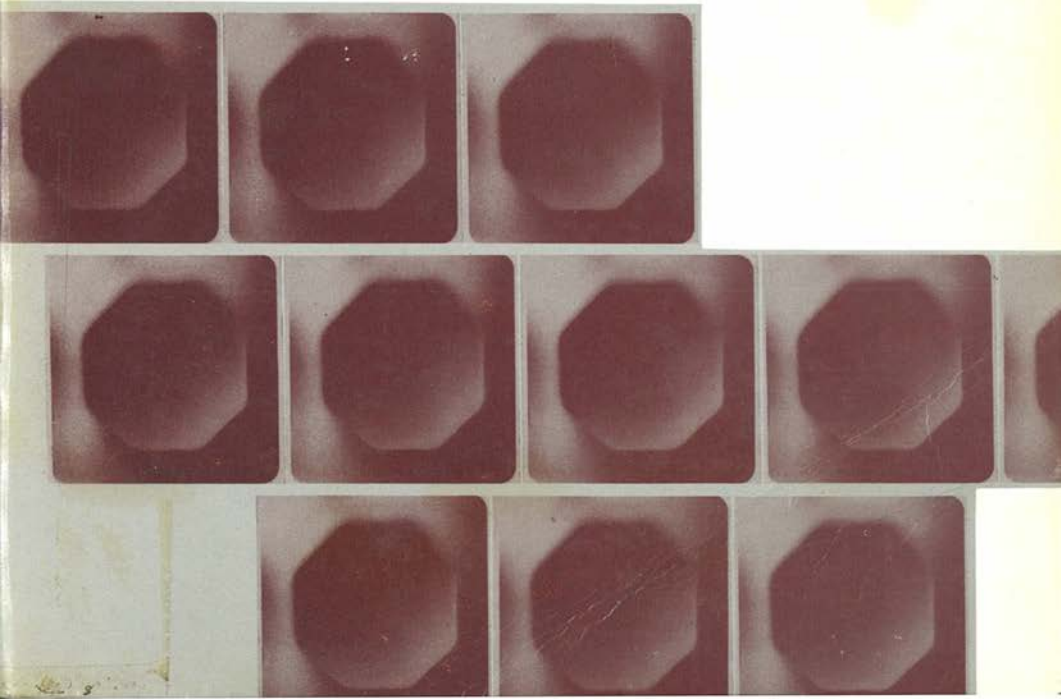
INTRODUZIONE AL PERSONAL E BUSINESS COMPUTING

EDIZIONE ITALIANA

RODNAY ZAKS

JACKSON ITALIANA EDITRICE

INTRODUZIONE AL PERSONAL E BUSINESS COMPUTING

di **RODNAY ZAKS**

Versione Italiana
di Lino Casamatti
e Roberto Mazzoni

JACKSON ITALIANA EDITRICE
Piazzale Massari, 22 - 20125 Milano

Grafica di Daniel Lenoury

Questo libro contiene informazioni complete e affidabili. Tuttavia la Sybex non assume alcuna responsabilità per l'uso del libro: nemmeno per violazione di brevetto o altri diritti di terze parti che ne risultino.
Non è garantita alcuna autorizzazione sottostante a brevetto o a diritti di brevetto dai costruttori di dispositivi. I costruttori si riservano il diritto di cambiare i componenti circuitali in qualsiasi momento senza alcun preavviso.
In particolare, le caratteristiche tecniche ed i prezzi sono soggetti a rapido cambiamento. I paragoni e le valutazioni sono proposti solo per il valore educativo e come i principi di guida.
Si rimanda il lettore ai dati dei costruttori per le specifiche esatte.

Prima edizione in lingua italiana: 1979
Stampato in Italia da Litografia del Sole - via Isonzo 14 - 20094 Buccinasco

INDICE

PREFAZIONE

Questo libro è stato scritto per il lettore che non conosce nulla sui computers. La sua comprensione perciò, non richiede una preparazione tecnica o elettronica.

È un'introduzione pratica, progressiva a tutti gli elementi di un sistema a computer reale. È stato ideato per una facile lettura. Il lettore con una mentalità tecnica troverà ulteriori informazioni nella sezione delle Appendici.

I primi tre capitoli vi introdurranno nel mondo dei microcomputers. Si parlerà di ROM e RAM, ed "userete effettivamente" un sistema, così da capire le funzioni dei suoi componenti.

Se volete capire come funziona, il Capitolo 4 vi porterà all'interno della scatola. Comunque, non è indispensabile.

Se volete programmarlo, potrete scegliere fra due tecniche: *linguaggio assembly* e *linguaggio ad alto livello.*

I Capitoli 5 e 6 vi introdurranno a queste tecniche. Il Capitolo 7 vi fornirà i requisiti specifici per l'uso del Business Computer, un'applicazione fondamentale dei nuovi sistemi microcalcolatori.

A questo punto avrete acquisito la conoscenza necessaria per analizzare le possibili scelte.

Una delle scelte più significative dal punto di vista del costo può non essere la scatola del microcomputer in sé, ma le periferiche. Il Capitolo 9 presenterà tutti i tipi di periferiche che possono essere usati e che debbono essere usati in circostanze specifiche, e perché.

Il Capitolo 10 presenterà i computers più importanti presenti sul mercato, e le loro caratteristiche.

Sapendo a questo punto cosa usare, la successiva domanda è: dovete comprare un sistema adesso? Il Capitolo 11 porrà questa domanda ed esaminerà le alternative per l'utente commerciale.

Le trappole usuali che aspettano l'utente commerciale sono molte, ed esse sono descritte nel Capitolo 12 ("Come Fallire").

Si può avere un po' di aiuto, ed esso è presentato nel Capitolo 13 ("Aiuto").

Infine, si può tentare una previsione del futuro su scala limitata, e nel Capitolo 14 ("Conclusione e Prospettiva") viene presentata una prospettiva.

CAPITOLO 1

L'ERA DEL MICROCOMPUTER

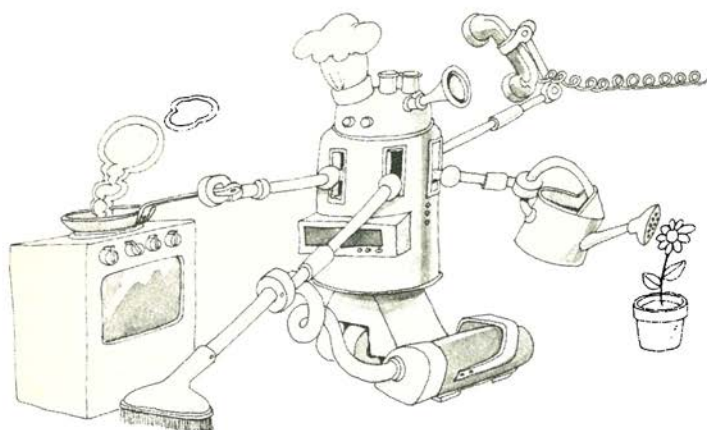
A casa

Sono le 7. L'allarme ronza sommessamente nella stanza. Jim si sveglia e blocca l'allarme. Un display simile a un cinescopio TV in miniatura s'illumina vicino al letto e mostra un messaggio. "Controlla il file RV prima di andartene". Jim si alza. Prima di lasciare la stanza egli preme due tasti sulla tastiera a lato "BR". Questo è un ordine di colazione che avvierà automaticamente la macchina del caffé e scalderà le frittelle dolci che Jim ama alla mattina. Appare la conferma sul piccolo display simile a un televisore accanto al letto. "CAFFÈ PARTITO. SARANNO SCALDATE DUE FRITTELLE DOLCI".

Jim ora procede nel suo studio così da verificare la lista degli appuntamenti, come richiesto. Mentre cammina attraverso la casa, le luci si accendono automaticamente, in sequenza, quando egli passa attraverso corridoi e stanze.

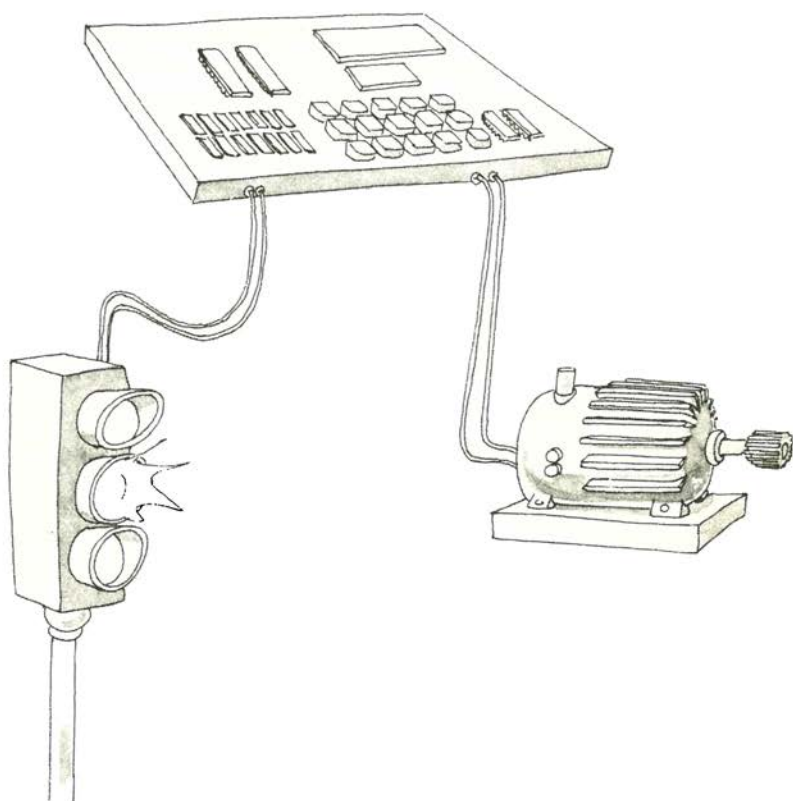
Jim raggiunge il suo studio. La lista degli appuntamenti "RV" è già presente sul grande terminale televisivo posto sulla sua scrivania. Essa elenca due nuovi appuntamenti che erano stati presi per lui in ufficio dopo che se ne era andato. Egli scopre che è stato lasciato dal suo superiore un messaggio importante durante la sera, che gli richiede di recarsi urgentemente ad un meeting commerciale speciale alle 9 di questa mattina. Jim fa una nota alla sua nuova lista di appuntamenti, e interroga nervosamente il suo home computer per scoprire chi altri prenderà parte al meeting commerciale. Sfortunatamente, la lista delle persone presenti al meeting, che è stata raccolta dal suo superiore, è protetta contro letture non autorizzate, e non può essere mostrata sullo schermo. Jim si alza e decide semplicemente di affrettarsi ed arrivare in ufficio al più presto. La sua colazione è già preparata e lo attende. Il riscaldamento della casa è stato acceso automaticamente e la stanza per la colazione è diventata piacevolmente calda.

Jim mangia in fretta, nervosamente, mentre sua moglie ed i figli continuano a dormire. Appena finito, egli preme la tastiera della cucina per preparare la sua automobile. La porta del garage si apre ed il motore dell'auto si accende automati-

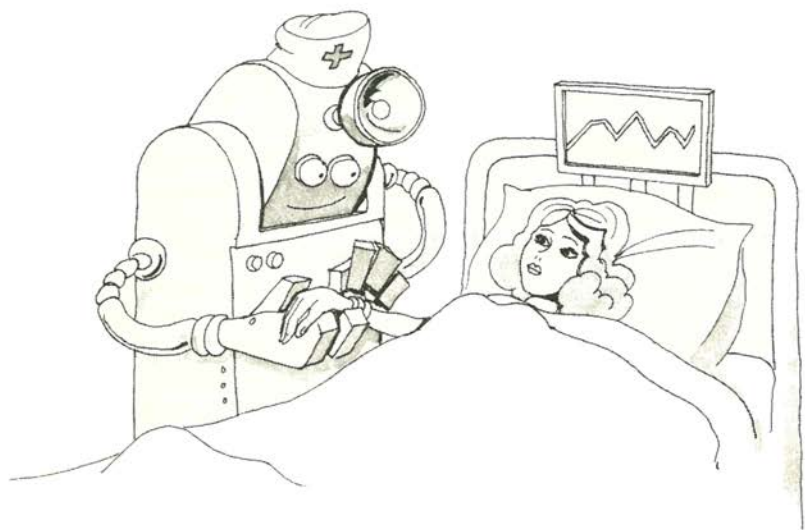
camente. Jim ritorna al terminale principale nel suo studio ed interroga rapidamente gli house computers di due suoi colleghi. Trova che essi hanno già timbrato il cartellino in ufficio. Decide di partire in fretta. Va al garage, dove il motore della sua auto è caldo ora, e si allontana. Le luci nella casa si spengono automaticamente appena egli esce. Mezz'ora più tardi, avverrà una sequenza simile quando sua moglie si sveglierà.

Jim è ora nella sua auto e vorrebbe arrivare in ufficio il più presto possibile. Egli preme la tastiera della sua auto per interrogare il computer principale per il traffico cittadino. È informato su una deviazione raccomandata attraverso una via trasversale così da diminuire il tempo necessario per arrivare al suo ufficio. Sfortunatamente, l'itinerario raccomandato non è familiare a Jim. Egli chiede assistenza. Il computer della sua auto interrogherà ora, i sensori stradali e visualizza automaticamente le svolte che deve fare per seguire il percorso desiderato. Il display della sua auto sta ora lampeggiando "volta a destra al prossimo incrocio". Jim segue le istruzioni. Egli controlla rapidamente il computer del contachilometri, che indica ancora 20 km da percorrere con un tempo stimato di viaggio di 22 minuti. Jim si sente meglio ora: arriverà in ufficio con un buon margine di tempo per il meeting. Egli non si deve preoccupare di alcuni ovvi disguidi. Il computer della sua auto ha già controllato il motore mentre si stava riscaldando, e ha stabilito che tutti i componenti funzionali erano in condizioni soddisfacenti. Il computer del contachilometri ha pure determinato che la quantità di carburante rimasta nel serbatoio sarebbe stata sufficiente a coprire la distanza usuale.

In ufficio

Jim ora arriva nel suo ufficio. Al suo arrivo, egli poggia il palmo della mano su una speciale cornice rettangolare equipaggiata con un sensore, e la porta si apre. Jim è stato identificato dal computer di sicurezza, che allo stesso tempo registra il suo ingresso. Sulla sua scrivania, Jim, trova uno stampato del computer che elenca i punti da trattare nel meeting commerciale delle ore 9,00. Reso consapevole dagli obiettivi del meeting, egli va immediatamente al suo terminale personale, dove ri-

chiede una copia di importanti elenchi di dati che egli vorrebbe avere a disposizione nel corso del meeting. Anche la lista di persone partecipanti è sbloccata ora, e Jim la può vedere sul suo schermo televisivo, e scoprire chi altro prenderà parte al meeting. Un aggiornamento lampeggia sullo schermo. Il meeting è stato rinviato di 15

minuti. Jim decide di chiamare Peter un collega fidato, e discutere i loro risultati. Contemporaneamente si accende un piccolo display sopra il suo televisore ed un messaggio entrante lampeggia "sta chiamando Mr. Gorvin della Precise System Inc." Egli schiaccia il pulsante di "rifiuto". La sua segretaria è stata automaticamente avvisata che lui non vuole rispondere alla chiamata in questo momento. La visualizzazione del messaggio entrante gli ha permesso di prendere una decisione immediata senza interrompere la conversazione col suo collega. Chiude la conversazione e va nella stanza del meeting.

All'ospedale

Durante questo tempo, sua moglie Linda, che attualmente lavora essenzialmente a casa come architetto, decide di visitare la sua parente Jane in mattinata, prima di iniziare il proprio lavoro. Si reca in auto all'ospedale. Qui, l'hostess scrive sulla tastiera del terminale di reception il suo nome, e il nome è accettato, il che indica che la visita è approvata, sia in termini dell'ospedale che in termini di approvazione del dottore per le visite. La camera, in cui la sua amica si trova, appare sul display. È la stanza 305 al terzo piano. Lei procede verso la stanza. Jane ha avuto un'operazione seria ed è sottoposta a monitoraggio automatico. Sonde speciali controllano

le sue funzioni vitali, come il ritmo cardiaco, la pressione del sangue, l'attività cerebrale. il ritmo respiratorio, la temperatura corporea, ed altre informazioni. Questi sensori sono collegati al computer al lato del letto che esamina continuamente i valori misurati confrontandoli coi limiti che sono stati stabiliti dal medico e controlla combinazioni di valori che potrebbero indicare una disfunzione del corpo. Finora tutte le indicazioni vitali sono state normali. Comunque, nella stanza accanto il programma di monitoraggio del cuore ha registrato un'aritmia che è probabile che accada prima di un effettivo attacco cardiaco. L'immediato segnale di pericolo che ha svegliato Jane, che ha sentito il medico affrettarsi nella camera e prendere immediatamente le precauzioni per prevenire il possibile collasso cardiaco. L'intervento ha avuto completamente successo, e l'attacco è stato prevenuto.

Jane spiega a Linda che quando lascerà l'ospedale, le condizioni del suo cuore dovranno essere ugualmente controllate per un periodo di almeno due mesi. Ella ha deciso, seguendo il consiglio del medico, di affittare un'unità portatile dall'ospedale. Questa unità le darà un segnale immediato, se una possibile disfunzione verrà prevista dal computer portatile. Inoltre alla sera, ella dovrà collegare il suo microcomputer portatile al telefono, permettendogli di trasferire i dati raccolti durante il giorno al computer dell'ospedale, che condurrà poi un sofisticato programma di analisi, così da diagnosticare sia progressi verso la guarigione che altri segni di complicazioni. Tutte le medicine di Jane non sono determinate dal computer a fianco del suo letto, ma sono determinate dal grande computer dell'ospedale che controlla tutte le prescrizioni dei medici per Jane confrontandole con la lista di altre medicine che lei sta già usando e con una lista di medicine alle quali potrebbe essere allergica. Ogni combinazione potenzialmente pericolosa è immediatamente segnalata all'attenzione dei medici che l'assistono.

Jane, che è un persona molto attiva, ha portato il suo microcomputer personale nella sua stanza così da poter mettersi a lavorare lentamente sulla relazione che ha preparato nelle settimane passate. Può usare il suo microcomputer per comunicare col computer della biblioteca, ed esaminare i contenuti di sezioni selezionate di libri che vuole consultare. Ella può quindi proseguire il suo lavoro per la relazione con tranquillità.

Ha l'avvertenza di non usare il suo microcomputer troppo a lungo poichè il suo dottore le ha vietato qualsiasi lavoro per periodi più lunghi di tre ore. Passate le tre ore, il suo personale computer viene automaticamente scollegato da un computer di montaggio.

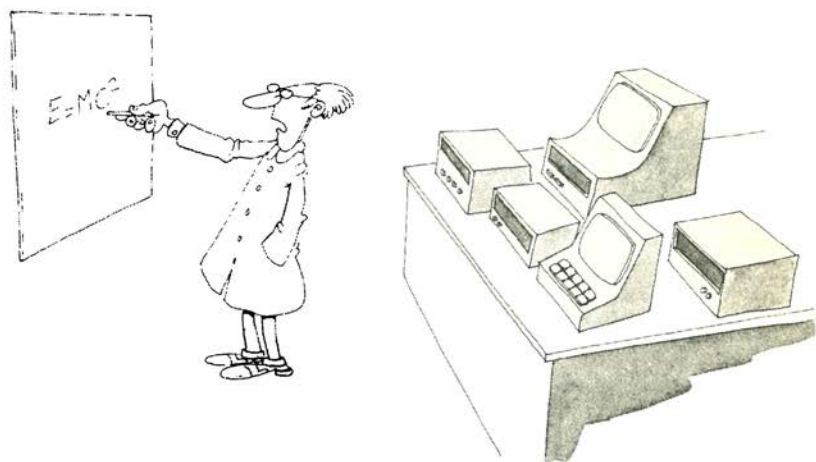
Di nuovo a casa

Linda sta tornando di nuovo a casa, dove intende prendersi cura rapidamente dei lavori domestici, e riprendere il suo lavoro di architetto. Tornata a casa, consulta il suo microcomputer personale per avere una lista di cose urgenti da fare per la giornata. Appare che lei deve trasferire denaro dai suoi conti di risparmio, poichè il

suo conto corrente è quasi esaurito. In aggiunta, alcune bollette domestiche scadranno entro un giorno. Per prima cosa decide di prendersi cura di questo. Collegatasi al computer bancario, ordina che sia fatto il trasferimento appropriato dal suo conto al conto dei suoi creditori. Per ogni transazione compone sulla tastiera un codice di identificazione speciale che verifica l'autore della transazione e la sua validità. Poi, si collega al computer del suo negozio preferito così da fissare la consegna di una lista di articoli. Comunque questa mattina è indecisa su quale verdura e frutta prendere. Richiede una panoramica della merce disponibile e una lista di prezzi. Fa poche annotazioni, e poi fissa il resto della lista di spesa. La merce sarà consegnata alle 4 del pomeriggio, come specificato nel rettangolo alla sommità del suo schermo televisivo. Ha ora preso cura dei suoi immediati bisogni per la giornata, e decide di lavorare per un'ora sulla sua lezione di lingue. Sta imparando lo spagnolo. La stanza è riempita dalla conversazione in spagnolo. I suoni sono generati da un programma, attraverso un sintetizzatore di voce. Allo stesso tempo sul terminale appare il testo scritto corrispondente. Le è chiesto di ripetere le parole. Ogni volta che fallisce nel dare un'imitazione soddisfacente del suono, il programma riprende con la frase precedente e la ripete finchè ottiene un'imitazione sonora soddisfacente, oppure dopo che lei ha fatto cinque tentativi. Il programma continua in sequenza, facendo eseguire a Linda una serie di esercizi. Dopo un'ora Linda decide, di smettere per questo giorno, di mangiare, ed incominciare a lavorare sul suo progetto architettonico. Prima di spegnere il terminale temporaneamente, per il periodo in cui pranza, ella fissa una lista di eventi che potrebbero innestare il sistema sonoro della casa automaticamente dentro la cucina. Quindi va in cucina. Più tardi, lavorerà nella sua stanza, disegnando le sue piante architettoniche su una tavola speciale, vedendoli mostrati sullo schermo, e correggendoli con l'aiuto del suo computer personale. Quando i disegni saranno sufficientemente completati, li trasmetterà al computer dell'ufficio dove saranno esaminati e criticati o approvati dal suo direttore. Più tardi nella serata, quando interrompe il suo lavoro, lo schermo si accende e su di esso lampeggiano due messaggi - due suoi vicini l'avevano chiamata nel pomeriggio, ma erano stati prevenuti dall'interrompere il suo lavoro. Avevano dovuto lasciare dei messaggi, che erano stati immagazzinati nella memoria del sistema e che ora vengono visualizzati.

I bambini torneranno presto da scuola, ed anche loro useranno il computer di Linda per i loro compiti a casa. Linda spera di comprare presto alla figlia più grande un proprio computer.

La città elettronica

Tutti i microcomputers nella casa di Jim, come pure in altre case, uffici, ospedali ed altri edifici sono interconnessi in una rete complessa. Le informazioni sono disponibili continuamente a qualsiasi ora ed istantaneamente, per tutti coloro che sono autorizzati ad accedervi ogni trasferimento di informazione è effettivamente istantaneo, e può essere attuato da qualsiasi terminale, nella casa, l'auto o l'ufficio.

I modelli di lavoro sono stati radicalmente cambiati. Molte meno persone lavorano in ufficio, a meno che essi debbano essere personalmente in contatto con uno solo dei loro colleghi, o usare risorse speciali disponibili solo in ufficio. Una grande quantità di tempo è spesa lavorando direttamente a casa o in qualche altro posto, tutte le mansioni possono essere eseguite con l'assistenza di un terminale. Tutti i processi che potevano essere eseguiti a macchina in casa o in auto, come pure sistemi di comunicazioni quali il telefono, o tutti i procedimenti che potevano essere automatizzati sono ora eseguiti e controllati da microcomputer in miniatura. Il vecchio orologio da polso è ora diventato un piccolo microcomputer equipaggiato con una tastiera in miniatura ed un terminale che può essere usato per comunicare con uno qualsiasi degli altri computer. Il lavoro è diventato più efficiente e creativo, traducendosi in una settimana lavorativa più breve.

Quando?

Lo scenario sopra era fantasia. Quando sarà realtà? *Tutte le attrezzature tecniche descritte sopra possono essere realizzate oggi.* Semplicemente per ragioni economiche, non sono state realizzate ancora su vasta scala, o non sono disponibili ad un costo sufficientemente basso; comunque si può sicuramente prevedere che un gran numero saranno realizzate nei prossimi dieci anni. Naturalmente ne saranno dispo-

nibili molte di più, che non si possono ancora immaginare. Lo scopo di questi paragrafi era di proporre la varietà di applicazioni ora possibili col progresso del microcomputer.

I microcomputers

I microcomputers sono il risultato di un progresso sorprendente nella tecnologia MOS (Metal Oxide Semiconductor) che rende possibile realizzare un computer completo su un piccolo rettangolo di silicio (un "chip" approssimativamente di 5 x 5 millimetri). Le funzioni che erano normalmente compiute da un grande computer che occupava un'intera stanza, possono essere eseguite ora da un computer a chip-singolo. Altre applicazioni, e tecniche di utilizzazione saranno presentate in questo libro.

Impariamo, ora, come si usa il computer di Jim o Linda al giorno d'oggi.

CAPITOLO 2

IMPIEGO DEL SISTEMA

INTRODUZIONE

Abbiamo appena acquistato un sistema a microcomputer: comprende un mobile microcomputer, un display di tipo televisivo con una tastiera, ed un normale registratore a cassetta. Naturalmente esistono molte scelte, e saranno passate in rassegna sistematicamente nei capitoli seguenti.

Lo scopo di questo capitolo è capire quali sono gli elementi principali di un sistema usandoli.

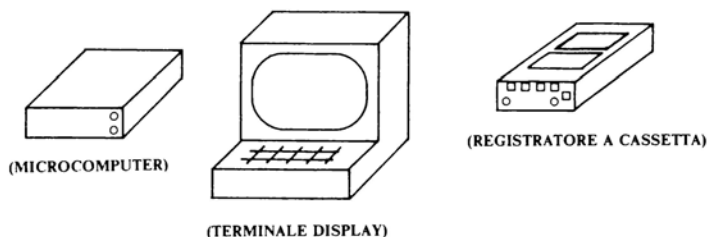



Fig. 2-1: Un sistema personal microcomputer.

COLLEGAMENTI

Siamo ansiosi di "fare qualcosa" col sistema. Insieme, connettiamo il sistema. Abbiamo tre moduli, più tre cavi. Il mobile del microcomputer eseguirà i programmi. Il display televisivo con la sua tastiera si chiama terminale CRT. CRT sta per tubo a Raggi Catodici (identico ad un cinescopio televisivo). "Terminale" indica che esso comprende una tastiera più alcune funzioni di processing (di procedura) richieste. Il registratore a cassetta è di tipo standard. Semplicemente questo ha un contatore ed un jack di controllo a distanza.

Per prima cosa colleghiamo il terminale CRT al microcomputer.
Il cavo standard di interconnessione ha come estremi i connettori "RS-232". Essi si inseriscono da una parte nel CRT, e nel microcomputer dall'altra estremità.

Quasi tutti i microcomputers e tutti i terminali CRT hanno questa interfaccia standard RS-232. Comunque, i microcomputers che hanno il display incorporato nel microcomputer possono non averne bisogno (per esempio il PET).

Il collegamento del registratore, richiede un cavo speciale. Viene inserito nell'ingresso del microfono, nell'uscita per l'altoparlante esterno e nella presa del controllo a distanza sul lato del registratore, e in una presa speciale sul fianco del computer.

Fig. 2-2: Questo sistema integra la tastiera nel mobile del microcomputer.

Alimentiamo tutte le unità:

- Il microcomputer ha un tasto on/off (acceso/spento). Lo accendiamo e appare la luce ON.
- Il terminale CRT si accende ed appare un quadrato bianco sullo schermo. Questo quadrato è un "cursore", esso indica la prima posizione dove il microcomputer visualizzerà un carattere sullo schermo. Noi premiamo "reset" sulla tastiera del CRT ed il cursore si sposta alla sommità sinistra dello schermo. Questa è la posizione dove noi (o il microcomputer) cominceremo a "scrivere sullo schermo".

Ora premiamo un tasto sulla tastiera. Il computer risponde immediatamente: "READY" (pronto) appare sullo schermo. Il sistema ora è pronto ad accettare co-

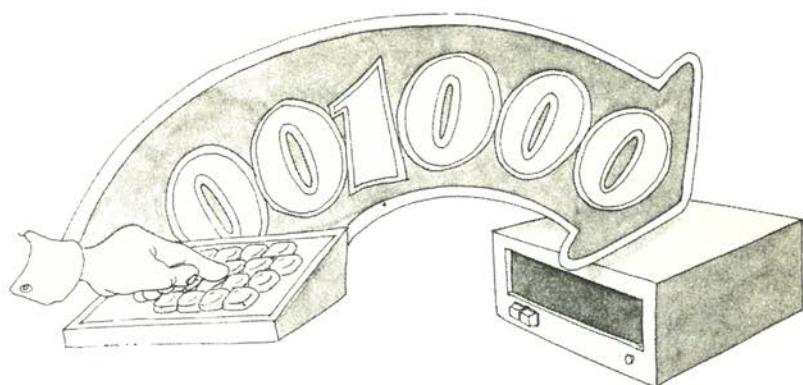



Fig. 2-3: Premendo un tasto sulla tastiera si invia un codice binario al microcomputer.

mandi da noi, ed aspetterà indefinitamente. Quando, noi premiamo il tasto sulla tastiera, la chiusura è rilevata dall'elettronica della tastiera, e convertita in un "codice binario" standard (chiamato ASCII, che sarà illustrato più avanti). Il microprocessore ha ricevuto un segnale, che richiedeva di leggere il carattere. Esso prontamente legge il carattere attraverso il cavo di interconnessione RS 232. In questo istante l'azione del computer non dipende dal carattere: tutto ciò che fa consiste nel visualizzare "READY". La parola "READY" è ritrasmessa al display come cinque caratteri consecutivi "R-E-A-D-Y". Sullo schermo osserveremo:

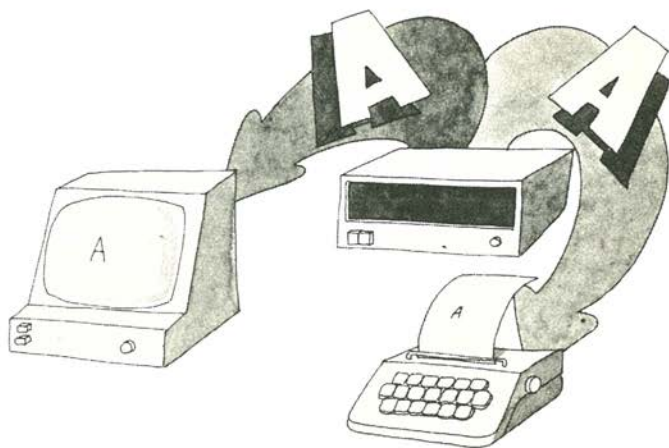



Fig. 2-4: Il microcomputer risponde.

- la lettera "A" che noi abbiamo appena premuto. Questa è chiamata "echo-back" (eco, ripetizione) dal computer.
- sulla linea sotto: READY

In realtà il computer ha inviato altri due caratteri al display, così che "READY" non apparisse sulla stessa riga di "A". Il primo carattere è chiamato "line-feed" (fornisci una nuova riga): il cursore si è abbassato di una riga. Il secondo carattere è chiamato: "carriage return" (ritorno del carrello, per carrello si intende figurativamente quello della macchina da scrivere), ed il cursore si è mosso alla posizione estrema alla sinistra dello schermo, giusto sotto la "A". Quindi è stato inviato "READY".

Questo "echo-back" è un'importante caratteristica: in una macchina da scrivere tradizionale, una lettera viene stampata quando si preme un tasto. In un terminale a computer sia una telescrivente che un display CRT, un carattere viene stampato o viene visualizzato dal computer. Ciò ha due vantaggi:

1 - Verifica di una trasmissione senza errori: vedendo apparire la "A" sullo schermo si è sicuri che è stata letta adeguatamente dal computer, e che il cavo di comunicazione funziona correttamente in entrambe le direzioni.

2 - Il carattere può essere interpretato. Per esempio, in un sistema commerciale, può essere richiesta una parola d'ordine prima che un utente possa usare il sistema. Supponiamo che la parola d'ordine sia "HENRY 8". Sarebbe ovviamente un errore avere la parola d'ordine mostrata sullo schermo quando viene composta sulla tastiera. Non ci dovrebbe essere alcun eco di una parola d'ordine. Qualsiasi buon sistema lo garantirà.

Ma lasciamo il display CRT e facciamo un gioco.

FARE UN GIOCO

Stiamo per caricare il sistema con un programma preso da una cassetta, per fare il gioco del tic-tac-toe col computer. È necessario caricare il programma nel sistema (un programma scritto nella memoria del microcomputer è volatile). Il suo contenuto sparisce quando si toglie l'alimentazione. Questo è uno sfortunato inconveniente della tecnologia LSI (large scale integration) usata per raggiungere questa micro miniaturizzazione e un basso costo. I programmi sono immagazzinati permanentemente su un dispositivo di memorizzazione magnetica come le cassette su cui non ha effetto il fatto che l'alimentazione sia inserita o disinserita. Dopo avere alimentato il sistema, il programma, che risiede sul disco, sarà trasferito nella memoria centrale del sistema, così da poter essere eseguito. L'utente, seduto alla tastiera batterà le istruzioni necessarie, in accordo col manuale d'uso per trasferire il programma chiamato "tic-tac-toe" (gioco del tris) nella memoria centrale del sistema. La sequenza effettiva dipende dal costruttore. Una volta che il tic-tac-toe è stato letto, non cambiamolo nè cancelliamolo.

Ora usiamolo. Il manuale allegato al programma tic-tac-toe spiega come usare il programma.

Noi componiamo sulla tastiera il comando "LOAD TCTO FROM TAPE", (carica il tic-tac-toe dal nastro), ed appare sul display CRT la configurazione di una tabella di nove quadrati. Ora siamo pronti a giocare. Dobbiamo introdurre i nostri movimenti. Introdurre le nostre mosse significa fornire dati al computer. Esso quindi risponderà. I dati in questo esempio sono introdotti specificando in quale quadrato vogliamo introdurre una "X" (in questo programma particolare il computer gioca sempre con uno "O", e noi abbiamo sempre la "X"). Andiamo a prendere il centro della tabella. Specifichiamo "2-2" che sono le coordinate della riga 2 e colonna 2. Terminiamo la nostra istruzione in input (ingresso) con un "carriage-return" (ritorno di carrello) uno speciale carattere analogo a quello sulla macchina da scrivere. Il carattere speciale è usato per dire al programma: "Io ho finito di battere sulla tastiera il mio input. Per favore procedi".

Questo programma è abbastanza veloce ed appena dopo aver "pensato" un secondo esso risponde. Un cerchio appare in posizione 1-1. Siamo pronti a rispondere .Il gioco ora procede nella maniera ovvia fino a che vinciamo noi o il computer. Il programma tic-tac-toe (gioco del tris) è un programma abbastanza semplice, e si può eseguire efficientemente, cioè rispondere rapidamente. Se volessimo giocare a scacchi, contro un programma "intelligente", la risposta del computer sarebbe molto più lenta poichè ha bisogno di eseguire un programma lungo prima di "sapere" cosa fare. Questo diventerà evidente più avanti.

A questo punto abbiamo illustrato le funzioni del *dispositivo di input* (la tastiera attraverso la quale abbiamo fornito i dati al computer). Abbiamo illustrato la funzione del computer che ha eseguito il programma tic-tac-toe, ha letto i dati che erano forniti, e ha mostrato sul display la risposta.

Abbiamo illustrato la funzione del terminale CRT quale *dispositivo di output* (uscita). È stato il dispositivo usato dal computer per mostrarci le mosse.

Infine è stato usato un nastro come dispositivo per immagazzinare a lungo termine programmi che non devono sparire quando l'alimentazione viene tolta.

Naturalmente se volessimo giocare un altro gioco, cioè eseguire un altro programma, potremmo battere sulla tastiera un'altra serie di comandi e caricare in memoria un programma differente, così potremmo giocare, per esempio, il gioco NIM.

Alla fine di questo particolare gioco, il computer visualizza: "WANT ANOTHER GAME?" (Vuoi un altro gioco?)

Rispondiamo "NO'.

Il computer visualizza "READY".

È tornato al modo "executive" (esecutivo) o "monitor" (monitoraggio) dove attende (indefinitivamente) i comandi dell'utente e non fa altro.

Proviamo le nostre abilità con la tavola pitagorica. Caricheremo un altro pogramma.

UN ALTRO PROGRAMMA

Per caricare il programma, noi battiamo sulla tastiera:

"LOAD MATH FRON TAPE" (carica la matematica dal nastro)

seguito da un "carriage return". Il carriage return è un segnale di "attenzione" per il microcomputer. Dice ad esso che il nostro messaggio è completo, e che può elaborarlo.

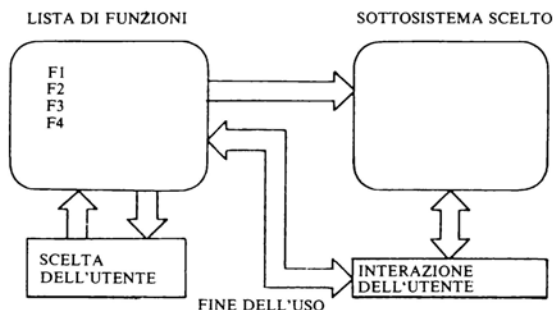



Fig. 2-5: Il sistema propone un "menu" di programmi.

Ma questa volta, non funziona:

"ERROR ABOVE. PLEASE REENTER"

(Errore sopra, per favore introdurre di nuovo)

Il computer accetterà solo comandi specificati. Il comando corretto è "LOAD... FROM...". L'errore di battitura FRON è stato rilevato automaticamente.

Si , un "computer" veramente intelligente dovrebbe capire che noi intendevamo "FROM", non "FRON". Sfortunatamente la complessità di accettare ordini parzialmenti non corretti è tale che nè i computer piccoli né quelli di media grandezza la fanno. Infatti ciò non vale il costo addizionale.

Così dobbiamo introdurre di nuovo il comando:

"LOAD MATH FROM TAPE" (Carriage Return)

La frase è accettata ora, ed il computer risponde:

"SEARCHING MATH..." (sto cercando la matematica)

Se avessimo notato l'errore di battitura immediatamente, noi avremmo potuto usare un'importante agevolazione disponibile sul nostro terminale CRT: esso esegue il "text-editing" (revisione del testo). Basta premere il tasto speciale "left arrow" (freccia a sinistra) ed il cursore si sposta a sinistra di un carattere. Lo facciamo

spostare fino a che si trova sopra la "N"di FRON, e noi battiamo M. M è sostituita al posto di N automaticamente. Poi muoviamo il cursore a destra battendo sul tasto speciale "right arrow" (freccia a destra) e procediamo.

Come minimo, ogni VDT (Video Display Terminal) deve permettervi di muovere il cursore in tutte le direzioni, così che gli errori possano essere corretti convenientemente (o dati addizionali introdotti).

Ma ritorniamo al nostro registratore a cassette. Premiamo "FAST FORWARD" (avanti veloce), ed esso incomincia a muoversi. Dover premere il tasto "forward" è un inconveniente che viene eliminato con periferiche più costose, quali i disks. Fermiamo il nastro a 550 sul contatore. Ora aspettiamo, ed alcune parole curiose appaiono sullo schermo:

NIM (Gioco, Schanghai)
CHEK (Checkers = Dama)
STOCK

Il microcomputer sta cercando nel nastro per il nostro programma. Alla fine la parola magica appare:

"MATH FOUND" (Matematica Trovata)

e dopo un certo ritardo:

READY-SPECIFY (Pronto specificare)
1-FOR MULTIPLICATION (Per moltiplicazione)
2-FOR DIVISION (Per divisione)
3-FOR ADDITION (Per addizione)
4-FOR SUBTRACTION (Per sottrazione)
ENTER YOUR CHOICE (Introducete la scelta)

La ragione per cui ci ha messo così tanto è che noi abbiamo fermato il nastro su 550 così da non "overshoot" (andare oltre) ed il microcomputer ha letto tutti i programmi intermedi prima di trovare il nostro programma "MATH". Ma ora l'abbiamo. Illustriamo alcune delle sue caratteristiche.

Questa volta il programma inizia col mostrare un "MENU" che offre quattro possibilità: moltiplicazione, divisione, addizione, sottrazione. Introduciamo: "1"

Domanda: *Questo è sufficiente?*

Risposta: *No. Dobbiamo battere sulla tastiera "carriage return" per dire al computer che abbiamo completato la nostra introduzione. In questo modo possiamo correggere gli elementi introdotti muovendo il cursore prima che vengano elaborati. Questo è un meccanismo di conferma.*

Ora abbiamo specificato che vogliamo un quiz moltiplicativo, e questo semplice programma stampa

12 x
23 =
―――――
?

Noi dobbiamo introdurre la risposta.
Bene, questo sarà lasciato come esercizio al lettore.
Riassumiamo la nostra esperienza finora.

SOMMARIO

Ora abbiamo usato la maggior parte delle funzioni di un sistema:
- La *tastiera* è stata usata per introdurre *comandi* o *dati.* La sua capacità di *editing* è stata usata per correggere errori muovendo il cursore.

- il *display* è stato usato dal computer per comunicare con noi. Esso visualizza solo caratteri, non immagini come un televisore regolare.
- il *registratore a nastro* immagazzina i programmi. È poco costoso, ma piuttosto lento. Nel caso di giochi, e brevi programmi, questo è accettabile.
- il *microcomputer* ha un programma built-in (incorporato) speciale, il *monitor* che controlla la tastiera costantemente e permette all'utente di usare il sistema per mezzo di caratteri di processo. In aggiunta, questo sistema ha un *sistema di file* che permette ad esso di recuperare *files* (programmi) simbolici come "MATH" dalla cassetta.

Infine, il microcomputer è equipaggiato con una *memoria interna*, dove immagazzina: 1 - il programma letto dalla cassetta, 2 - i dati che noi battiamo sulla tastiera.

Ora introdurremo tutte le definizioni convenzionali così che si possa descrivere componenti e funzioni accuratamente.

CAPITOLO 3

DEFINIZIONI DI BASE

Applicazioni del microcomputer

Non è possibile limitare il numero delle possibili applicazioni dei microcomputers. Comunque, per lo scopo di questo libro, è conveniente classificare le applicazioni dei microcomputers in tre ampie categorie; applicazioni personali, applicazioni commerciali, applicazioni industriali.

Le *applicazioni personali* dei microcomputers sono caratterizzate da sistemi a basso costo, usati nell'ambiente domestico. Questi sistemi sono destinati a fornire sia divertimento che servizi su misura per i loro utilizzatori. Si vedrà che le loro caratteristiche tecniche sono sostanzialmente differenti da quelle dei microcomputers delle altre due classi di applicazioni.

I *business microcomputers* sono usati nell'ambiente dell'ufficio. Essi sono usati per compiti commerciali come libro paga, gestione d'inventario, acconti pagabili, ricevibili. Si vedrà che i requisiti di questi microcomputers commerciali li rendono i più costosi sistemi a microcomputer di tutte le categorie. Questo è dovuto non solo ai loro sostanziali requisiti di elaborazione, ma più che altro ai costosi terminali necessari per una prestazione soddisfacente.

I *microcomputers industriali* sono microcomputers installati in applicazioni industriali per controllo di processi. Microcomputers incorporati nel sistema telefonico domestico, nell'ospedale, in edifici pubblici, ascensori ed automobili, ricadono in questa categoria. Tali sistemi non saranno discussi qui.
Per una discussione dettagliata, il lettore viene indirizzato al nostro libro, Microprocessors: From chips to Systems disponibile entro il 1979 nella versione italiana della Jackson Editrice.

Al fine di differenziare queste applicazioni, sarà necessario usare un certo numero di termini tecnici. È giunto il momento di definirli.

Anche il lettore non tecnico è vivamente invitato a leggere la definizione di questi termini, se vuole comprendere cosa sarà presentato nei capitoli seguenti.

Definizioni di base

Un *microcomputer* è un computer la cui central processing unit (unità di elaborazione centrale) è realizzata con un microprocessore a chip singolo.

Questi termini tecnici saranno definiti nel resto di questo capitolo. In breve, un microcomputer è un computer che è realizzato con componenti LSI.

LSI sta per Large Scale Integration. LSI è il risultato dell'evoluzione dell'elettronica contemporanea tendente alla microminiaturizzazione dei componenti. Dal transistor discreto degli anni che hanno seguito la II Guerra Mondiale, il progresso nella tecnologia ha permesso la realizzazione di un numero sempre maggiore di transistor su un singolo pezzo di silicio. Ora è possibile realizzare circuiti con fino a 50.000 transistori su un tale "chip". Il *chip* è il più piccolo pezzo rettangolare di silicio, su cui è realizzato il circuito. La tecnologia è progredita dai transistori discreti alla SSI (Small Scale Integration) alla MSI (Medium Scale Integration) o alla SLSI (Super Large Scale Integration) (con densità di più di 500.000 transistori per chip).

A questo punto, diventa necessario definire la struttura base del computer così da definire gli elementi funzionali che costituiscono un microcomputer.

Un *computer* può essere semplicemente definito come un dispositivo calcolatore per scopi generici capace di eseguire un *programma.* Un *programma* è una sequenza di istruzioni. Le istruzioni tipiche manipolano informazioni contenute in un dispositivo di *memoria.* Le *istruzioni aritmetiche* effettuano operazioni come addizione, sottrazione, o talvolta moltiplicazione e divisione. Le *istruzioni logiche* realizzano operazioni logiche come un "AND logico", un "OR logico". In aggiunta, un'istruzione di "branch and test" (diramazione e test) permette che siano eseguite differenti porzioni del programma a secondo delle condizioni da esaminare. Il concetto di programma sarà descritto più avanti.

Qualsiasi computer comprende tre elementi funzionali di base:

1 - La Central Processing Unit (Unità di Elaborazione Centrale)

La CPU è incaricata di prendere le istruzioni dalla memoria, e di eseguirle. Tipicamente, essa comprende una memoria interna molto veloce destinata ad aumentare la velocità dell'esecuzione delle istruzioni: i *registers"* (registri). I registri trattengono l'informazione per l'elaborazione da parte della CPU.

2 - La memoria

La memoria è destinata ad immagazzinare tali programmi e i dati su cui essi operano. Può essere usato un certo numero di dispositivi di memoria, che saranno descritti più avanti in questo capitolo (ROM, RAM).

3 - Input-Output (Ingresso-Uscita)

I dispositivi di input-output permettono al computer di comunicare col mondo esterno. Un dispositivo di input-output tipico può essere una tastiera attraverso la quale i dati vengono introdotti nel sistema. Un dispositivo di output tipico è una

stampante, o un display CRT, attraverso i quali i dati possono essere visualizzati o trasmessi al mondo esterno.

L'insieme di componenti fisici del sistema è chiamato *hardware*. L'insieme dei programmi è chiamato *software*. Un termine intermedio viene usato per i programmi che risiedono in un tipo speciale di memoria chiamata *read-only memory* (memoria a sola lettura), (ROM) che non possono essere cambiati.
Poichè questi programmi (software) sono realizzati su un componente hardware fisso, sono *chiamati firmware*.

Un *microprocessore* è semplicemente la realizzazione di una central processing unit in un componente singolo.

La parola *chip* ora è usata non solo per designare un pezzo rettangolare di silicio che contiene il circuito, ma il componente stesso (chiamato il "package" cioè il contentitore).

Si dovrebbe ricordare che queste sono definizioni semplificate.
Infatti, nella pratica tutti i primi microprocessori erano solo CPU "quasi complete". Di conseguenza essi richiedevano un certo numero di circuiti esterni per fornire un CPU effettivo. In particolare, circuiti addizionali erano spesso richiesti (e spesso lo sono ancora): un oscillatore esterno (il clock), un cristallo (per un preciso riferimento) e drivers per i bus del sistema(*bus* verranno spiegati più avanti. Essi sono semplicemente elementi di comunicazione per il sistema).

In aggiunta ai tre elementi funzionali suddetti, un *microcomputer* comprende di solito l'alimentatore, necessario per fornire le tensioni necessarie per tutto il sistema, ed un contenitore. Se la memoria del sistema è piccola, un sistema a microcomputer (eccetto l'alimentatore ed il cabinet) può essere ora realizzato su un single board, cioé su una sola scheda. Comunque, nei sistemi general purpose (per scopi generali) è desiderabile permettere un'espansione successiva come l'aggiunta di più memoria, o più dispositivi esterni.

Dispositivi complessi esterni tipicamente richiedono *schede di interfaccia o controllers*. Per questa ragione, la maggior parte dei microcomputers general purpose hanno un aspetto comune.

Un microcomputer oggi ha l'aspetto di una piccola scatola (le reali dimensioni dipendono dal costruttore, ma tipicamente hanno le dimensioni di un cassetto). Tale contenitore del microcomputer contiene la scheda della CPU, una o più schede di memoria, più una a più schede I/O (Input/Output) progettate per fornire l'interfaccia standard con le periferiche comuni. In aggiunta comprende l'alimentatore richiesto ed un bel contenitore. Contrariamente ai computer tradizionali, molti contenitori di microcomputer non sono dotati di pannello frontale. *Un pannello frontale* è un pannello equipaggiato con lampadine ad interruttori sulla parte frontale del contenitore progettata per esaminare i programmi a livello hardware. Si vedrà più avanti che una tale attrezzatura non è indispensabile su un sistema di personal o business computer. Poiché aumenta notevolmente il costo del computer, non viene quasi mai fornito.

In breve, un microcomputer tipico è un contenitore che contiene due o tre schede di logica e l'alimentatore, che si comporta come un computer.

Ora vengono fornite in pratica assieme ad ogni microcomputer standard due *periferiche* standard. Il dispositivo di *input-output* standard è una tastiera, essen-

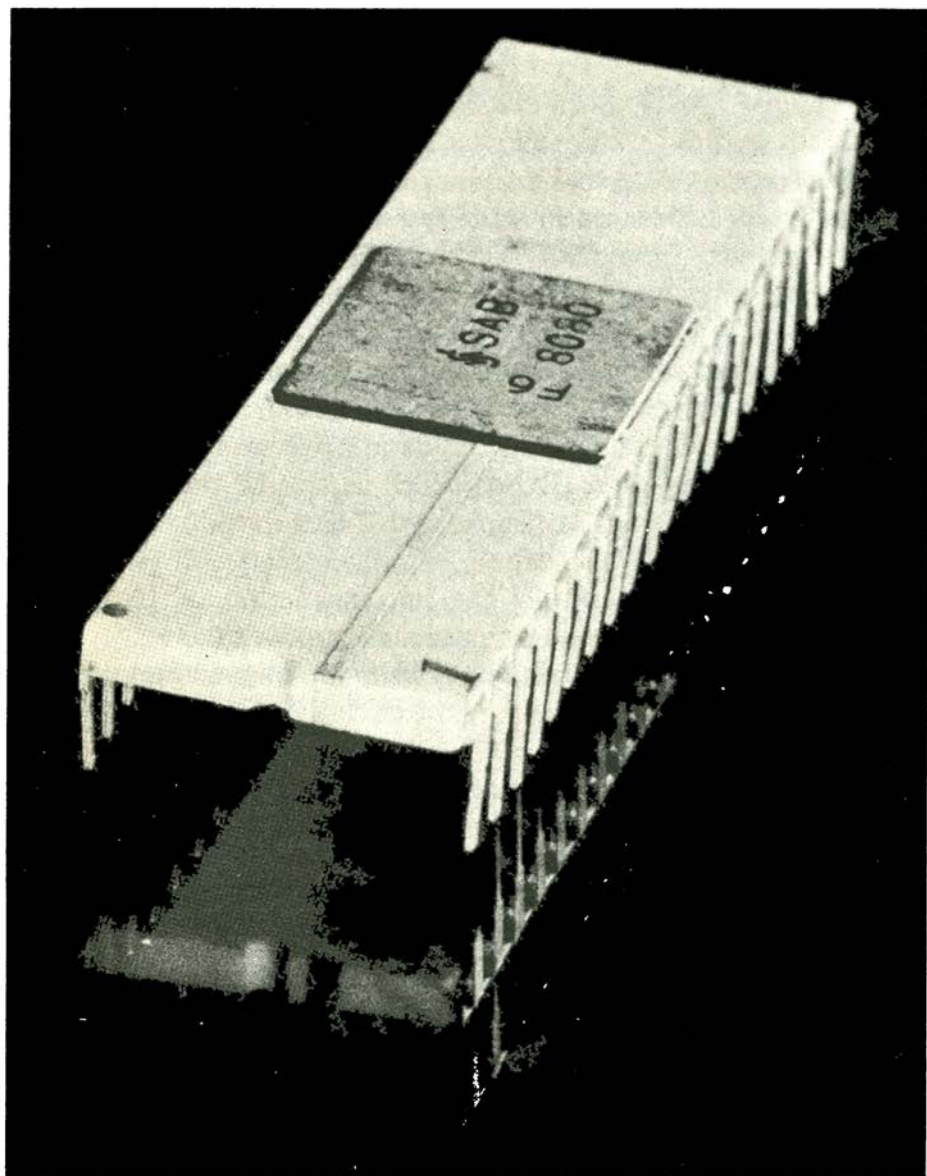



Fig. 3-1: Un microprocessore (8080). Il contenitore ha 40 piedini.

zialmente analoga ad una tastiera di macchina da scrivere. Ogni volta che l'utente preme un tasto sulla tastiera, il carattere è codificato in codice binario (un insieme di 0 e 1) e trasmesso al microcomputer.

Un dispositivo di *output* standard, ora fornito con i microcomputer, è il display CRT. Il display CRT è semplicemente, un monitor televisivo senza i soliti strumenti di sintonizzazione e ricezione. E' semplicemente ideato per visualizzare sullo schermo caratteri generati dal microcomputer.

Poiché sia i programmi che i dati hanno bisogno di occupare una memoria molto più grande della memoria standard fornita nel sistema a microcomputer, è necessario equipaggiare il sistema microcomputer con una memoria di massa addizionale più economica. Esistono due alternative essenziali. La più economica sta nell'usare un registratore a cassetta. Un registratore a cassette ordinario può essere usato per immagazzinare programmi o dati. È economico, ma limita drasticamente le prestazioni del sistema.

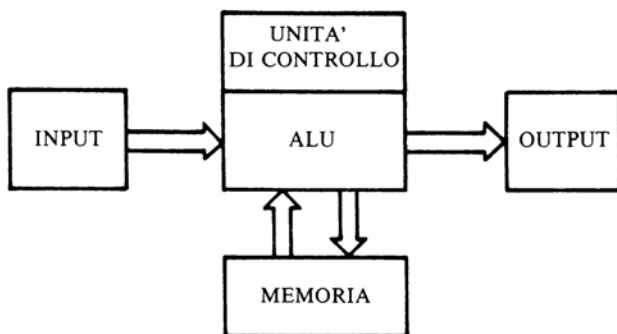



Fig. 3-2: I cinque elementi funzionali di ogni computer.

Il *floppy disk*, che sarà descritto in dettaglio più avanti, è diventato il dispositivo di memoria di massa standard per microcomputer. È notevolmente più costoso di un registratore a cassette, ma è il miglior mezzo di memoria di massa disponibile oggi, che fornisca un rapidò accesso al suo contenuto ad un costo ragionevole. In aggiunta, si dovrebbe notare che per qualsiasi applicazione pratica, che manipola schedari, è necessario usare almeno *due* floppy disk. In questo modo si possono comparare due files o mescolarli insieme, oppure uno può essere letto mentre l'altro viene scritto. Una tale attrezzatura è indispensabile per un file processing (elaborazione dei file) conveniente. Questi concetti verrano studiati più tardi in maggiore dettaglio.

Naturalmente, a secondo dell'applicazione considerata possono essere necessarie periferiche addizionali o differenti. Comunque, a questo punto possiamo dire

che un sistema a microcomputer standard è formato da un contenitore del microcomputer (che realizza la CPU, più memoria ed interfaccia I/O) più una tastiera d'ingresso, più un display CRT per l'uscita, più un disk drive (attrezzatura per leggere ed incidere i disk) doppio (per immagazzinamento di massa). Questa è una "configurazione minima". In pratica si dovrà aggiungere al sistema almeno una stampante. Questa configurazione standard verrà usata come punto di riferimento. Si vedrà che un sistema più piccolo può essere preso in considerazione per applicazioni o usi personal. Si vedrà anche che, nella maggior parte dei casi, si dovrà considerare un sistema più grande per usi o applicazioni più complessi.

Costituzione di un microprocessore

Come è costruito un microprocessore? È stato indicato che i microprocessori sono solo un esempio particolare di componenti LSI. Un componente LSI tipico è realizzato su un *chip* vincolato ad un "*dual in line package*" (package a doppia linea) (un DIP). Un'illustrazione di un DIP appare in Fig. 3-4. Un DIP avrà tipicamente da 18 a 40 piedini, attraverso i quali il chip (dentro il DIP) può comunicare col mondo esterno.

Questa tecnologia è il risultato di un'evoluzione dei MOS. MOS sta per "metal oxide semiconductor". Il semiconduttore è il *silicio*. Il silicio quando viene *drogato* con impurità (fosforo o boro) diventa positivo o negativo (tecnicamente esso avrà eccesso di elettroni o deficienza di elettroni = *lacune*). *L'ossido* è ossido di silicio, usato come isolante, sulla cima del substrato di silicio. Il *metallo* è il gate (porta) del transistor che è realizzato tramite la deposizione sulla superficie dell'ossido di alluminio oppure di silicio.

Il processo di fabbricazione è relativamente semplice, nel suo principio. Un singolo cristallo di silicio viene accresciuto con grande cura, dando origine ad un reticolo cristallino perfetto. Questo cilindro di silicio (del diametro di quattro, cinque o sei pollici) viene quindi tagliato in fette molto sottili (chiamate *wafer)*. I wafer vengono *politi* ed assomigliano a specchi rotondi. Essi sono talmente sottili che sono molto fragili, e si rompono come il vetro. I chips saranno creati sulla superficie di questi wafer di silicio. *Su ogni wafer saranno creati tipicamente da 100 a 500 chip.* Un chip misurerà in effetti circa 5 × 5 millimetri su questo wafer. I transistor ed altri componenti vengono creati sul chip diffondendo impurità selettivamente all'interno del silicio depositando poi uno strato di ossido e depositando uno strato di metallo sopra l'ossido. Viene usato un processo di incisione (chimico) selettiva per definire le aree dove vengono eseguite la diffusione, l'ossidazione o la metallizzazione. Il processo è esattamente analogo alla stampa di illustrazioni in un libro. È un processo *fotolitografico*. Viene fissata una maschera sulla superficie del wafer di silicio, e l'emulsione fotografica (posta in precedenza sul wafer) viene esposta alla luce, sviluppata ed incisa chimicamente. Diffusione, ossidazione e metallizzazione, avverranno nelle aree incise del silicio. Quando tutti i "chip" saranno stati creati sul wafer, il wafer sarà *inciso* con punteruoli e rotto (lungo le lineee di incisione) in *chip* individuali. Ogni chip viene quindi montato in package individuali, che saran-

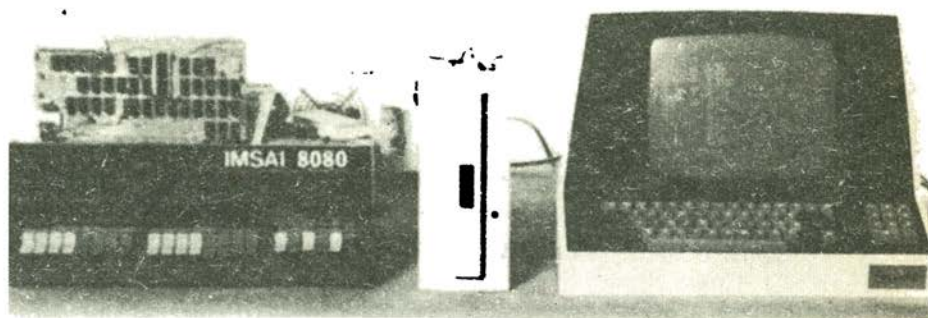



Fig. 3-3: Questo sistema comprende da sinistra a destra: 1 - mobile del microcomputer, 2 - il disk, 3 - il terminale CRT.

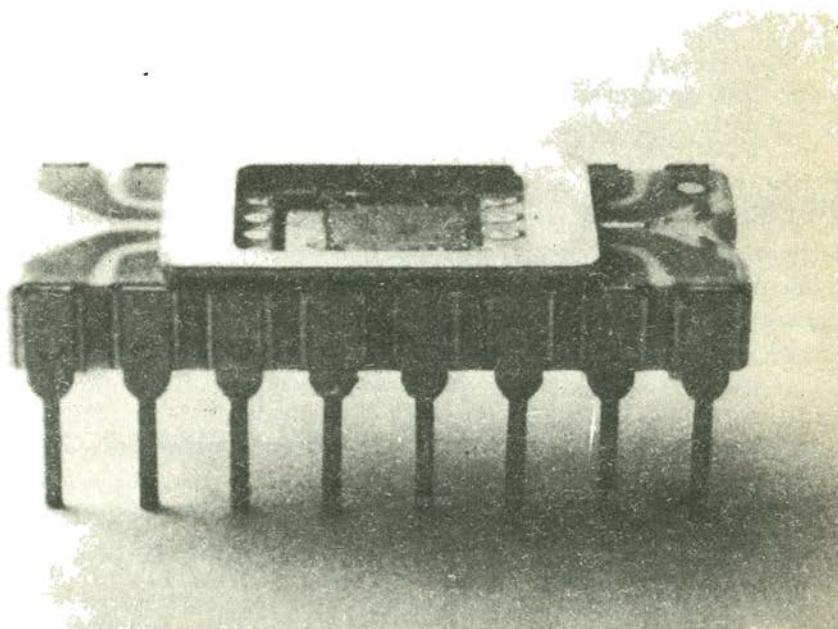



Fig. 3-4: Questo DIP è stato aperto per mostrare il chip.

no poi sigillati e diventeranno DIP (Dual in Lime Package, come quello di Fig. 3.4).

Dopo il chip di *memoria*, il *microprocessore* è stato uno dei primi componenti LSI standard ad essere introdotti (1971). Da allora è stato introdotto un gran numero di altri componenti LSI, ed in pratica tutti i componenti necessari a realizzare un microcomputer completo sono ora disponibili in forma LSI. Grazie al numero molto alto di transistors che possono essere realizzati su un solo chip, si è potuto realizzare un sistema a computer completo su di una singola scheda. E' diventato possibile realizzare un computer completo, benchè limitato su un *sinoglo chip.* Questo è il *"microcomputer ad un chip".* Comunque i microcomputers ad un chip sono limitati per quel che riguarda la quantità di memoria che esse possono avere come pure per il numero di linee input-output che essi possono avere e, per questa ragione, essi sono esclusivamente utilizzati per applicazioni di controllo. Essi non sono ancora utilizzabili come personal e business computers. In futuro, intere schede di microcomputer saranno ridotte ad un microprocessore ad unico chip. Comunque ciò avverrà tra diversi anni.

Esaminiamo ora un po' più in dettaglio ognuno dei tre elementi funzionali di un sistema a microcomputer. È importante comprendere la funzione di questi elementi, per capire la necessità che i vari dispositivi usati li realizzino come pure per capire i loro vantaggi. Si vedrà che ci sono poche "soluzioni ottimali" che sono indipendenti dall'applicazione considerata. Come per più complessi sistemi, il sistema a computer *deve essere scelto in funzione della applicazione voluta. Perché si possa fare una scelta buona e intelligente è sicuramente necessario capire cosa fa ognuno degli elementi funzionali.* Questo può essere paragonato alla scelta di qualsiasi sistema ad alta complessità, da un'automobile ad un congegno. È indispensabile capire le funzioni dei componenti così da valutare le possibili offerte dal mercato a meno che non si debba tener conto del prezzo. *Questo vale per l'utente commerciale.*

LA CPU

CPU sta per Central Processing Unit. La CPU andrà a prendere, decodificherà ed eseguirà istruzioni. Essa prende le istruzioni dalla memoria in cui sono immagazzinate. Le esegue per mezzo di una arithmetic logical unit (ALU unità aritmetico-logica) in cui sono eseguite le operazioni aritmetiche e logiche. La ALU è tipicamente equipaggiata con "registers" (registri) interni che forniscono memorizzazione ad alta velocità per dati usati di frequente. Alla fine, le istruzioni sono decodificate e sequenziate internamente da un'elemento speciale nella CPU chiamato Control Unit (CU, unità di controllo).

Tutte queste funzioni sono realizzate tipicamente su di un chip singolo, il chip del microprocessore. In pratica, può essere necessario un certo numero di componenti di supporto sulla scheda, come il clock ed il suo cristallo, che fornisce una frequenza di riferimento stabile. Componenti addizionali sono pure tipicamente necessari per amplificare i segnali: essi sono "drivers". La maggior parte dei microprocessori, oggi è capace di elaborare dati di 8 bit simultaneamente ("bit" sta per "bi-

nary digit", "cifra binaria" cioè uno "0" logico o un "1"). Un microprocessore a 8 bit è un microprocessore che può operare su 8 bit simultaneamente cioè un byte di dati (un *byte* equivale ad 8 bit).

I bus

Il microprocessore riceve e trasmette dati al e dal mondo esterno per mezzo di otto linee. Queste otto linee usate per trasmettere i dati sono chiamte *data bus* (bus di dati). Un bus è semplicemente un insieme di linee, raggruppate per funzionare. In questo caso è un data bus. Si dice bidirezionale, perché i dati fluiscono in entrambe le direzioni, al o dal microprocessore.

Per specificare da dove vengono i dati, o dove stanno andando bisogna fornire al microprocessore un numero di identificazione, che specifica l'origine o la destinazione dei dati. Questo è detto l'"*address*" (l'indirizzo). Usualmente viene fornito un *address bus* (bus di indirizzi) a 16 linee. Esso permette di indirizzare un gran numero di locazioni (esattamente 65.536). In gergo di computer si dice solitamente "64 K" invece di 65.530. Un K rappresenta $1024 = 2^{10}$.

Qui si può ricordare brevemente che un data bus ed un address bus, non sono ancora sufficienti per un funzionamento completo del sistema. A causa della natura dei vari componenti elettronici nel sistema, è necessario fornire *segnali di sincronizzazione*. I segnali per il trasferimento ordinario di informazioni lungo il data bus del sistema vengono forniti dal *control bus* (bus di controllo). Esso è il terzo bus standard connesso al microprocessore. Lasciamo i bus per adesso, e torniamo ai componenti hardware principali del sistema.

La memoria

Si è visto che la memoria del sistema immagazzina i programmi ed i dati su cui essi operano. Una memoria è organizzata in *words* (parole). In un sistema a 8 bit le parole sono lunghe 8 bit. In questo caso una parola è lunga appena un byte. In un sistema a 16 bit, una parola è lunga 2 byte. In un sistema a 4 bit una parola è lunga 4 bit (si dice "nibble" per 4 bit). Una *parola* non specifica un dato numero di bit in tutti i casi. Una parola è proprio l'unità di informazione logica su cui il microprocessore opererà. In un sistema tradizionale ad 8 bit accade che una parola sia uguale ad un byte.

Un esempio di byte (immagazzinato nella memoria sarebbe: "00000000" (8 zeri). Si vedrà, nella sezione della programmazione, che tutti i caratteri, tutti i dati, e tutte le istruzioni nel sistema sono rappresentati solo per mezzo di gruppi di 8 bit. Talvolta, si deve usare più di una parola per un prezzo di informazioni multi-word (multi-parola). Comunque, non è ancora conveniente usare un altro sistema elettronico diverso da quello basato sulla rappresentazione binaria, in cui, cioé, gli stati logici sono sempre "0" e "1". Per questa ragione, il sistema binario viene usato universalmente nei computers digitali.

Una rappresentazione più completa sia dei programmi che dei dati dentro la memoria del sistema sarà descritta nel capitolo della programmazione.

L'ideale sarebbe che l'intera memoria del sistema fosse economica, veloce e ampia. Sfortunatamente velocità e dimensioni non sono ancora compatibili con un costo basso. Per questa ragione sono universalmente usati *due tipi principali* di memoria: la *memoria principale* e la *memoria ausiliaria*, o "*mass storage*" (memoria di massa). La memoria principale di un sistema è contenuta nella scatola del computer. È realizzata con componenti MOS LSI.

Il mass storage è di solito un supporto magnetico come un "floppy disk" o una cassetta magnetica, che sono relativamente economici ed offrono una vasta capacità di immagazzinamento. Comunque essi sono più lenti della memoria principale.

I vari tipi di componenti usati per la realizzazione della memoria principale saranno descritti nel prossimo capitolo. Ne vengono usati essenzialmente due tipi. La read only memory (ROM, memoria a sola lettura), e la random access memory (RAM, memoria ad accesso casuale). Sfortunatamente, una memoria MOS LSI che può essere sia letta che scritta è volatile (la RAM). Il suo contenuto scompare ogni volta che viene tolta l'alimentazione elettrica. Questo può essere accettabile per dati temporanei, ma non è accettabile per programmi. Per questa ragione, è necessario immagazzinare un programma permanente in un tipo diverso di memoria la OM (Read Only Memory) che non può essere scritta (in cui cioè non si possono introdurre dati) ma che non è volatile.

Questi due tipi di memorie (ROM e RAM) sono sempre necessarie in un sistema a microcomputer standard. Nel caso di sistemi personal e commerciali, i programmi del costruttore risiedono nella ROM, ed i programmi dell'utente risiedono nella RAM, durante l'esecuzione. Quando non vengono eseguiti, i programmi dell'utente risiedono su di un disk o una cassetta.

Si vedrà che la dimensione di memoria richiesta dipende dall'applicazione considerata. Una dimensione tipica potrebbe essere da 2K a 8K di ROM ed almeno 4K di RAM. Questi numeri saranno discussi più tardi.

Input-Output

Il dispositivo universale di input è la tastiera alfanumerica. La tastiera alfanumerica è semplicemente la tastiera usata in un macchina da scrivere, spesso equipaggiata con tasti addizionali, che permette all'utente di specificare tutti i caratteri, più i numeri, più simboli speciali. La tastiera normalmente è equipaggiata con *encoder* (codificatore) speciale, che fornisce il codice a 7 o 8 bit direttamente al microcomputer, ogni volta che viene premuto un tasto.

È stato trovato che esso è il mezzo di input più efficiente per un operatore umano. Naturalmente in ambienti specializzati, possono essere usati altri dispositivi di input. Per esempio nei giochi TV possono essere usati per giocare una leva di comando o altri pulsanti speciali ed interruttori. Essi sono più economici, e adatti all'ambiente ma più limitati nella loro capacità di trasmissione di informazioni.

Il display CRT e la stampante sono diventati i due dispositivi universali di output per un microcomputer. Tecnicamente non è indispensabile avere *entrambi* i dispositivi di output. Una stampante per esempio è sufficiente. Comunque una stampante affidabile è molto più costosa di un display CRT ed ha anche lo svantaggio di essere relativamente lenta quando è necessario esaminare un testo lungo. Perciò per ogni personal o business microcomputer viene fornito un display CRT per avere un'informazione veloce e silenziosa. È necessario, nella maggior parte dei casi, fornire una stampante separata se sono necessarie registrazioni permanenti.

Poiché i dispositivi di input-output in genere sono dispositivi meccanici o elettronici complessi, ognuno di questi dispositivi necessita di un *controller* (controllore) separato, realizzato da una scheda che riceverà i comandi generati dal microcomputer o l'informazione di stato dal dispositivo, e li decodificherà. Leggerà pure l'informazione dal dispositivo, e li decodificherà nel formato richiesto per il microcomputer. Nella maggior parte dei casi, non è possibile collegare il microcomputer direttamente ad un dispositivo di input-output complesso: devono essere forniti un *controller o schede di interfaccia* per effettuare il collegamento al sistema.

Un sistema a microprocessore

Un *sistema a microprocessore* è essenzialmente la scheda completa ad eccezione dell'alimentatore, del contenitore, e delle periferiche. Una "*architettura*" di un tale sistema completo appare in fig. 3.5. Sulla sinistra, appare un microprocessore, che effettua la funzione CPU. Subito dopo a sinistra, appaiono i due tipi di memoria, la

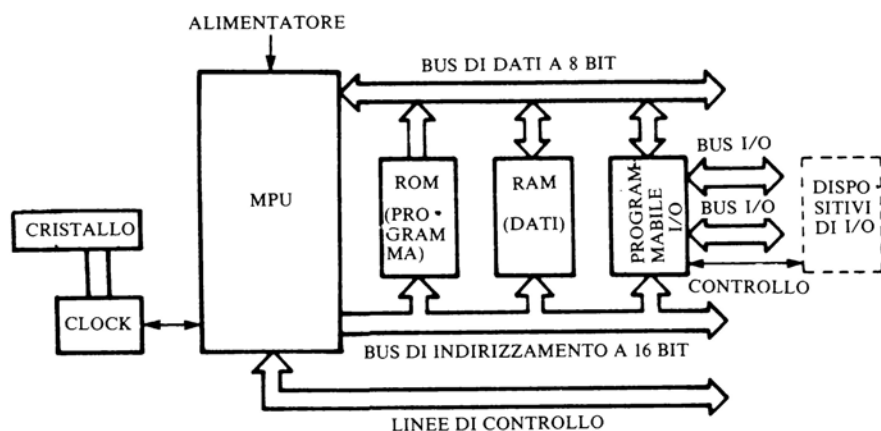



Fig. 3.5: Sistema microprocessore standard.

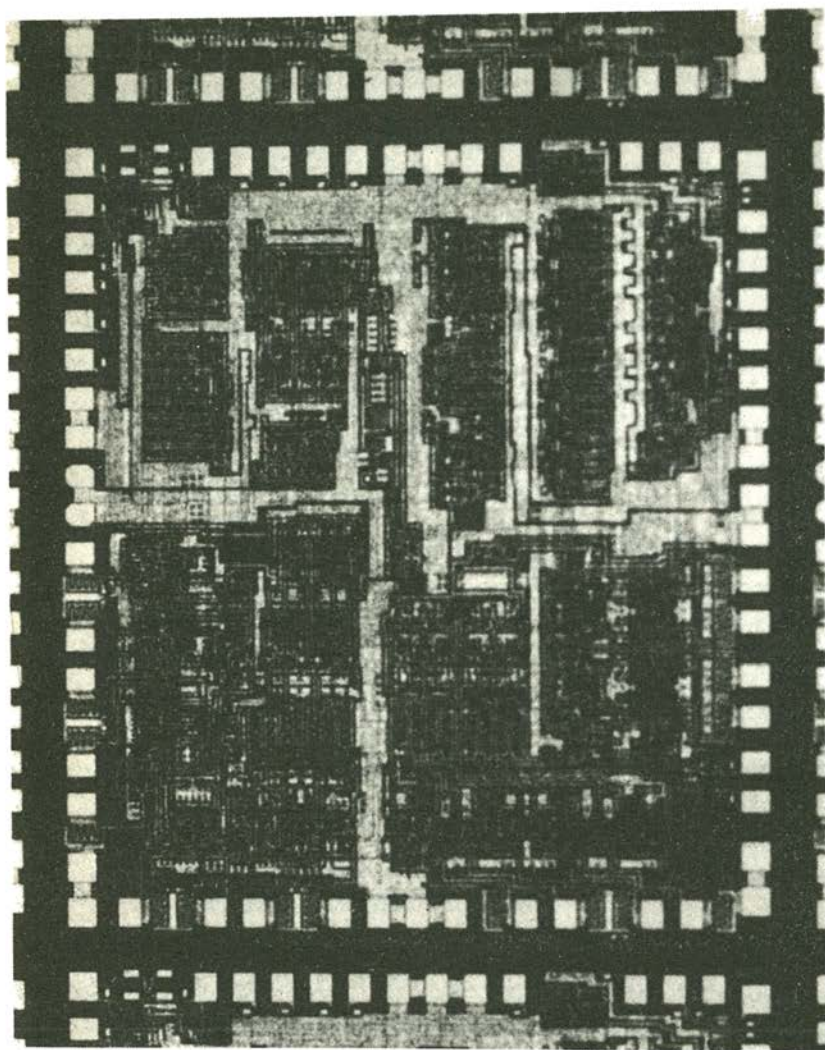



Fig. 3-6: Un "chip" sul wafer è circondato da altri chip.

ROM e la RAM. Poi c'è l'interfaccia di I/O, che può comunicare direttamente con le periferiche. Le tre larghe piste orizzontali sono i tre bus del sistema.

Sulla sinistra, appaiono il clock del sistema ed il quarzo che sono tipicamente esterni all'unità del microprocessore. L'interfaccia I/O crea, in aggiunta, un *bus I/O* che si collega direttamente alle periferiche. Sono stati fatti diversi tentativi di standardizzazione per fornire facili connessioni delle periferiche esterne ai bus di un sistema. Sarà mostrato, in particolare quel bus standard che è diventato dominante nel mercato degli hobbisti, il bus S-100. Sarà presentato in un capitolo seguente.

Il sistema a microcomputer

Il sistema a microcomputer è un sistema basato su un microprocessore completo ed usabile. Esso include tutti gli elementi funzionali richiesti, e altro. Come minimo, esso comprende la scatola del microcomputer (che contiene una o più schede, più l'alimentatore, più una tastiera ed un display CRT o una stampante).

Microcomputer o minicomputer?

È stato sempre difficile fornire una buona definizione di cosa sia un minicomputer. Per la maggior parte degli utenti, un minicomputer è il contenitore di un computer equipaggiato con almeno 4K parole di memoria (solitamente di 16 bit) ed una telescrivente (dispositivo input-output simile ad una macchina da scrivere), che si vende per meno di $ 10.000.

In pratica, un minicomputer è una versione ridotta del tradizionale grosso computer. Il suo processore è meno potente, ma risiede su una o due schede di circuiti logici, e questo ha come conseguenza un costo notevolmente ridotto, si è trovato che i minicomputer hanno molto successo quando vengono usati nell'ambiente scientifico ed in applicazioni commerciali limitate.

Qual'è la differenza tra il minicomputer ed il microcomputer?
In pratica non c'è nessun'altra differenza. Un microcomputer è stato definito come un computer la cui CPU è realizzata da un chip microprocessore. Un microcomputer è caratterizzato dal fatto che i suoi componenti sono componenti LSI cioè microminiaturizzati. A causa della difficoltà di costruire CPU altamente complesse su di un chip singolo, il potere di elaborazione di un microprocessore è stato limitato finora. Per questa ragione finora i microcomputer hanno avuto una capacità operativa alquanto inferiore al minicomputer tradizionale. Comunque, a causa del loro basso costo, i microcomputer hanno introdotto un'alternativa significativa ai minicomputers tradizionali.

Infatti la maggior parte degli utenti non hanno mai avuto bisogno dell'intera potenza del minicomputer, ma avrebbero voluto un prezzo molto più basso (esattamente ciò che il microcomputer ha portato). Come risultato ora anche i progetti di *minicomputer* includono generalmente i microprocessori. Almeno nelle versioni

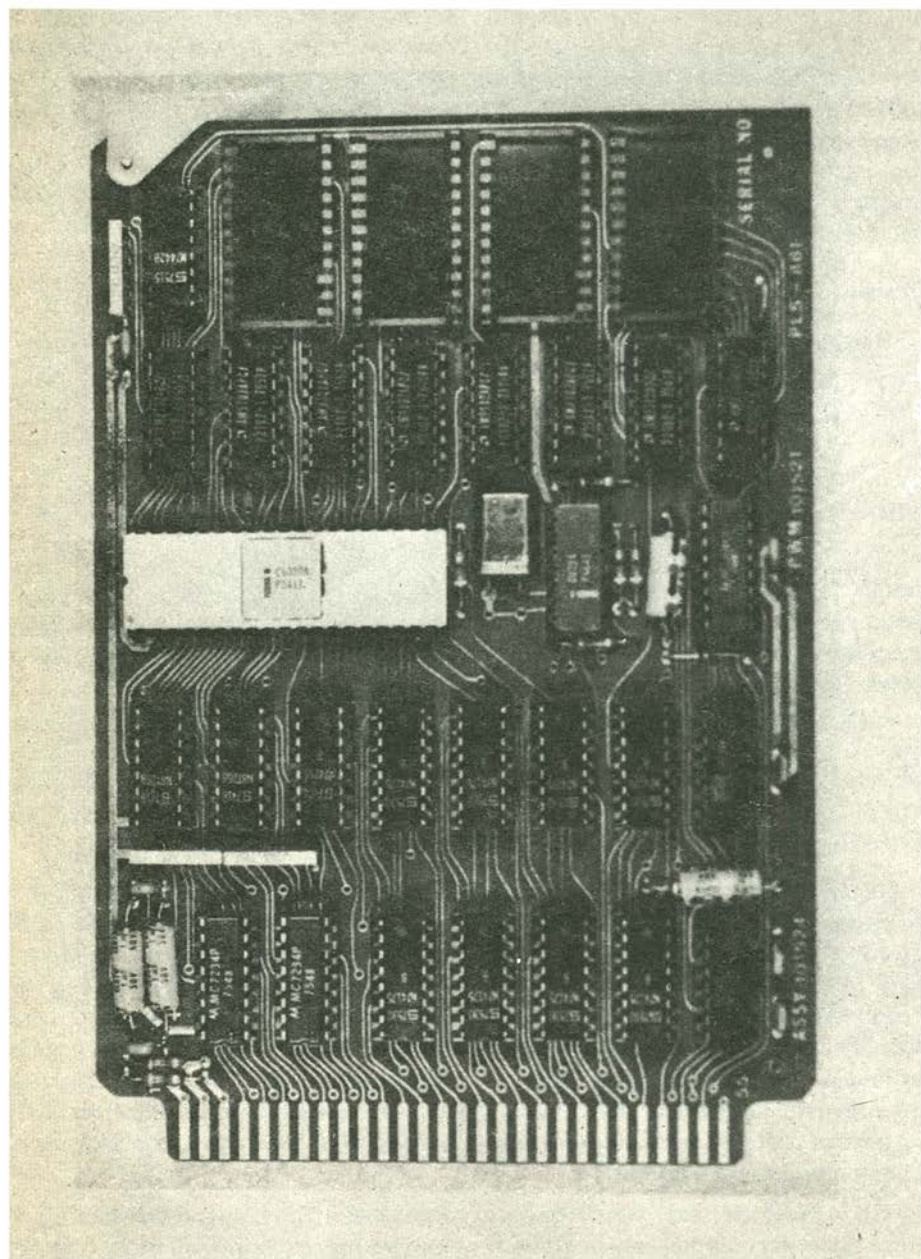



Fig. 3-7: Un microcomputer a scheda singola: il grosso chip chiaro al centro è il microprocessore.

più lente dei minicomputer tradizionali i microprocessori sono usati per realizzare la CPU. In altre parole, *i minicomputer più lenti sono oggi minicomputer.* Poiché la potenza dei microprocessori aumenta costantemente, si può prevedere che la maggior parte dei minicomputer sarà alla fine microcomputer (o viceversa). Non c'è più una rigida barriera tra "micro" e "mini". Comunque tra le estremità dello "spettro" esistono ancora differenze.

Un microcomputer generalmente è ancora un processore che ha un instruction set (gruppo d'istruzioni) piuttosto debole (limitato), che opera piuttosto lentamente, e con solo 8 bit alla volta. In più, i suoi bus di comunicazione esterna sono ridotti dal piccolo numero di piedini del DIP del microprocessore (tipicamente 40 piedini).

Per contro, un minicomputer ha un potente instruction set ed un bus ampio cosicchè può trasmettere più segnali simultaneamente (una lunghezza tipica della parola per un minicomputer è 16 bit). In breve un minicomputer opera tipicamente in modo più veloce ed ha più istruzioni di un microcomputer, avendo una parola di ampiezza doppia: un minicomputer tradizionale è ancora considerevolmente più veloce che il microcomputer *medio.*

Un'altra importante differenza sta nel fatto che i minicomputers esistono da un numero di anni maggiore. Come risultato, i minicomputers più venduti sono disponibili con una vasta libreria di programmi (software) che la maggior parte di microcomputer ancora non ha. Comunque il divario si sta colmando rapidamente grazie al mercato degli hobbisti. Questo punto sarà illustrato più avanti in maggiore dettaglio.

Per applicazioni scientifiche relativamente potenti, un minicomputer veloce è ancora desiderabile. Comunque, per il numero molto ampio di nuove applicazioni che non richiedono una potenza di elaborazione notevole, *il microcomputer è l'ideale.* In ogni caso non è una panacea.

Un microcomputer non può gestire efficientemente tutte le applicazioni. Si vedrà che esistono serie limitazioni, riguardo al possibile uso del microcomputer in applicazioni commerciali.

Esempi di minicomputer che sono diventati microcomputer sono: il Nova della Data General (il Nova/2) o l'LSI 11 (che realizza in LSI, il PDP 11/03).

Vantaggi dei microcomputers

I vantaggi dei microcomputers sono quelli della tecnologia LSI e del microprocessore a chip singolo. Essi sono tre:

1-Piccole dimensioni

Un microcomputer completo e funzionante può ora essere realizzato in un volume molto piccolo. Quando la quantità della memoria e delle funzioni input-output può essere limitata, si può persino realizzare un microcomputer completo in un

chip singolo. Ora si può realizzare facilmente nel volume di una borsa per documenti un sistema completo, la cui capacità di elaborazione è paragonabile ad un minicomputer lento. Infatti, in alcuni sistemi, quando lo spazio disponibile è sufficiente, lo stesso microcomputer risiede nel contenitore della tastiera.

2-Costo molto basso

Il costo di un tipico chip MOS è di pochi dollari. Un sistema completo richiede pochi chip ed il contenitore del microcomputer completo costa poche centinaia di dollari, a seconda della quantità di memoria che contiene. Il prezzo continuerà ancora a diminuire.

3-I vantaggi nella programmazione

Un microcomputer realizza per mezzo di programmi ciò che si era soliti realizzare con mezzi meccanici o altri mezzi. I vantaggi dei programmi sono la flessibilità con cui essi possono essere scritti, progettati, debugged (corretti) o modificati.

SOMMARIO

Ora sono state introdotte tutte le definizioni di base. Alla fine di questo capitolo vengono presentati degli esercizi per scopi di auto-test (verifica da sè). E' indispensabile capire chiaramente queste definizioni basilari per capire la maggior parte dei capitoli rimanenti.

A causa delle molteplici alternative sia meccaniche che elettroniche, ogni sistema differirà per la quantità di memoria, dispositivi di input-output, e come packaging. Comunque tutti i sistemi che verranno descritti hanno la stessa architettura, cioé gli stessi elementi funzionali. Una loro valutazione richiede la comprensione di queste funzioni.

Per il lettore con una mentalità maggiormente tecnica, il prossimo capitolo consentirà di dare un'occhiata all'interno della scatola, per vedere ciò che accade.

ESERCIZI PER AUTO-TEST

(Risposte nella pagina seguente)

3.1 - Quale dei seguenti elementi risiede normalmente nel contenitore del microcomputer: 1 - scheda della CPU 2 - scheda addizionale di memoria. 3 - scheda del disk controller. 4 - scheda speciale d'interfaccia, 5 - alimentatore, 6 - decodificatore della tastiera.

3.2 - Qual'è la differenza tra un minicomputer ed un microcomputer.

3.3 - Definisci un "bus".

3.4 - Qual'è la differenza tra una ROM ed una RAM?

3.5 - Ogni sistema ha bisogno di una RAM?

3.6 - E' indispensabile una mass-storage (memoria di massa)?

3.7 - Quali sono i tre elementi funzionali di ogni sistema a computer?

3.8 - Quanti bit ci sono in un byte? (1) 4, (2) 8, (3) 16.

3.9 - Quanti bit ci sono in una parola? (1) 4, (2) 8, (3) 16

3.10 - Cos'è un address?

3.11 - Sono dati i seguenti termini? (1) Caratteri (2) Numeri

RISPOSTE

3.1 - *1-2-3-4-5. Il decodificatore della tastiera è sotto la tastiera stessa. Naturalmente, se la stessa tastiera è integrata nel contenitore del microcomputer, allora anche (6) è nel contenitore.*

3.2 - *vedi il testo*

3.3 - *vedi il testo*

3.4 - *vedi il testo*

3.5 - *Si, altrimenti non ci potrebbe essere né input né output, poiché i caratteri devono essere immagazzinati nella RAM.*

3.6 - *No, se i programmi sono corti, e se l'utente ha voglia di batterli sulla tastiera ogni volta.*

3.7 - *CPU, memoria, input/output.*

3.8 - *8 bit*

3.9 - *Indefinito. Questo dipende dal microprocessore*

3.10 - *L'address è un numero che contrassegna la locazione in cui alcuni dati possono essere trovati.*

3.11 - *Si. Si.*

CAPITOLO 4
COME FUNZIONA

INTRODUZIONE

L'obiettivo di questo capitolo è spiegare in maggiore dettaglio le funzioni fornite dai componenti di un sistema a microcomputer. Questa informazione tecnica è necessaria a tutti coloro che intendono valutare le capacità tecniche e le deficienze tecniche di un sistema per un'applicazione specifica. È utile anche per l'utente che prende in considerazione un'applicazione commerciale ma "non gli importa sapere come funziona". Tuttavia coloro che non sono concretamente interessati al funzionamento tecnico di un sistema possono saltarlo. Esso fornisce al lettore interessato una spiegazione di come funziona un computer internamente e cosa fanno i vari componenti. Il capitolo precedente era solo una introduzione di base. Questo è un capitolo tecnico.

L'ARCHITETTURA BASE DI UN SISTEMA

I tre elementi funzionali di base di qualsiasi sistema computer sono la Central Processing Unit (CPU), la memoria, l'input-output (ingresso-uscita).

Poiché non è ancora possibile realizzare un numero sufficiente di transistors su un unico chip, ognuna di queste funzioni è attualmente realizzata da uno o più componenti hardware (i chips). Ora esamineremo in maggiore dettaglio queste tre funzioni e la loro realizzazione per mezzo di componenti specifici.

La Central Processing Unit

L'unità di elaborazione centrale (CPU) del sistema del computer è incaricata di prendere (dalla memoria), decodificare ed eseguire le istruzioni. Queste istruzioni sono contenute nella *memoria* del sistema. Un esempio pratico mostrerà come un'istruzione viene presa dalla memoria, portata nella CPU, decodificata ed eseguita.

Per tutti gli scopi pratici stabiliamo qui che la CPU sia realizzata da un chip microprocessore (chiamato comunemente MPU microprocessor unit, unità microprocessore). Questo non è completamente corretto: nella maggior parte dei sistemi possono essere necessari un *clock* ed un *cristallo* esterni e talvolta componenti addizionali. Questo sarà sufficiente per il nostro esempio. La fig. 4-1 mostra il microprocessore alla sinistra e la memoria sulla destra.

Un *programma* viene eseguito istruzione per istruzione. Normalmente, ogni istruzione viene eseguita dopo quella precendente. Queste istruzioni sono contenute nella memoria, l'*indirizzo* della istruzione successiva da eseguire è contenuta in un registro interno al microprocessore chiamato program counter (PC) (contatore di programma), così da mantenere traccia della localizzazione di un'istruzione nella memoria.

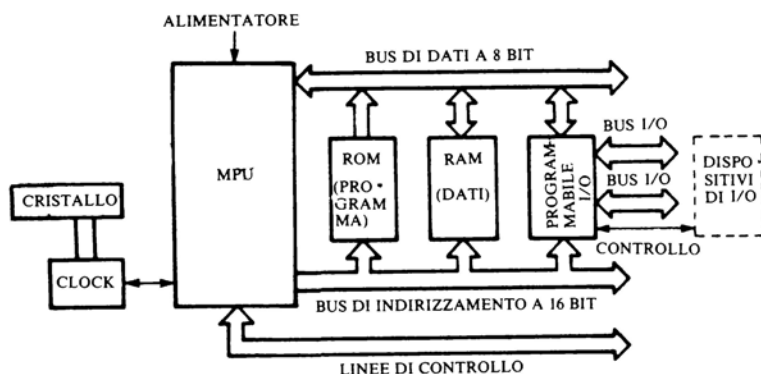



Fig. 4-1: Il sistema standard.

Un *registro* è semplicemente una piccola memoria interna che può contenere una o più parole di informazioni. Nel nostro esempio supponiamo che il *program counter* sia un registro a 16 bit. In altre parole questo registro può memorizzare 16 bit di informazione binari (degli 0 e degli 1). Questi 16 bit rappresentano l'indirizzo binario della prossima istruzione da prendere dalla memoria. Ora dovrebbe essere semplice prendere la prossima istruzione dalla memoria: il contenuto del program counter sarà "depositato" sull'*address bus* del microprocessore. Il nostro microprocessore standard è equipaggiato con 16 piedini chiamati *address pins* (piedini degli indirizzi), che permettono la preparazione di un indirizzo nel mondo esterno. 16 linee porteranno questo indirizzo alla memoria. *Queste 16 linee sono l'address bus del sistema del microcomputer.* Poiché, di solito, una memoria può sia leggere che scrivere, solitamente sarà necessario mandare un *segnale di lettura* (alla memoria). Il microprocessore genera il segnale, e la memoria legge il contenuto dell'indirizzo specificato. Questo contenuto, per definizione, è la prossima *istruzione* che deve essere eseguita dal microprocessore. Passeranno poche centinaia di nanosecondi (un nanosecondo è 10^{-9} secondi), prima che i dati provenienti dall'interno della memoria siano fisicamente disponibili. Questo breve ritardo è chiamato *tempo di accesso* della memoria. In un sistema a microcomputer standard, la memoria è larga 8 bit. *In risposta ad un indirizzo di 16 bit, essa andrà a prendere una parola, e*

questa parola è larga solamente 8 bit! È importante sottolineare che non c'è relazione tra il numero di bit forniti sull'address bus, ed il numero di bit che escono dalla memoria (in questo esempio 8 bit).

Un esempio simile sarebbe un indirizzo di una via. Dando un indirizzo di una via ad una persona, questi trova una casa o un edificio. Il fatto che l'indirizzo della via sia grande o piccolo, non ha nessuna relazione con l'effettiva dimensione dell'edificio che vi si troverà.

Potrebbe essere una piccola casa, o potrebbe essere un grosso edificio con più appartamenti. *Il dato è totalmente scollegato dall'indirizzo.* L'indirizzo è semplicemente la locazione dei dati all'interno della struttura che nel nostro caso è la memoria o la città.

Questa istruzione a 8 bit deve essere riportata al microprocessore, così che possa essere eseguita. Tutti i dati nel sistema transitano normalmente in un bus specializzato, il data bus (bus dei dati). La memoria è equipaggiata con 8 collegamenti alle 8 linee del data bus. Per cui l'istruzione a 8 bit è inserita nel data bus ed appare sulla destra dell'illustrazione. I dati viaggiano lungo il data bus e verso il microprocessore a destra dell'illustrazione. Essi sono portati all'interno di uno speciale registro interno del microprocessore chiamato instruction register (*registro delle istruzioni*) (IR). Questa volta il registro è largo 8 bit, poichè deve contenere solo 8 bit. Questo registro serve a trattenere l'istruzione successiva da eseguire. Quando arriva il momento di eseguire l'istruzione, gli 8 bit di questa istruzione saranno decodificati dal *decodificatore,* e sarà generato automaticamente il segnale di controllo interno appropriato dentro al microprocessore che porterà all'esecuzione dell'istruzione.

Il meccanismo per prelevare (fetch) un'istruzione dovrebbe essere chiaro ora. Come si preleverà la prossima istruzione? C'è un meccanismo automatico speciale: il program counter è equipaggiato con un *incrementatore* ("più 1"). Questo incrementatore incrementa di 1 il contenuto del program counter, ogni volta che viene usato. In questo modo automaticamente si avrà come risultato che ogni volta che si depositerà nuovamente il contenuto del program counter sull'address bus, si accederà al prossimo indirizzo che in sequenza segue quello precedentemente raggiunto. Abbiamo proprio ora costruito un *meccanismo di sequenza automatico* che andrà automaticamente a prelevare istruzioni dalle locazioni di memoria successive.

Naturalmente, ci sono casi in cui l'esecuzione di un programma non deve essere sequenziale. Si ha allora una istruzione di *branch* (diramazione) o go to (vai a...). In questo caso, l'istruzione speciale di *branch* modifica esplicitamente il contenuto del program counter così da forzare un *jump* (salto) ad una locazione differente. Ma questo è il soggetto di un altro libro (Programmazione).

Una domanda potrebbe ancora restare nella mente del lettore tecnicamente orientato: com'è eseguita fisicamente l'istruzione all'interno della CPU? Formiamo qui una breve descrizione. Per una descrizione completa si rimanda il lettore al prossimo volume di questa serie C201 - Microprocessori disponibile anch'esso entro il 1979 nella versione italiana curata dalla Jackson Editrice.

La sezione della *Control Unit* (unità di controllo) della CPU va a prendere e decodifica le istruzioni. L'esecuzione dell'istruzione è realizzata dalla sezione ALU della CPU, che è normalmente equipaggiata con *registri interni* specializzati. In figura 4-2 appare un'illustrazione dell'ALU tipica. I rettangoli verticali sulla sinistra dell'illustrazione sono i registri. Nel caso del nostro microprocessore "standard", questi registri conterranno 8 bit. Il simbolo a V sulla destra dell'illustrazione è la *arithmetic logical unit* (ALU) (unità aritmetico-logica). La ALU è incaricata di

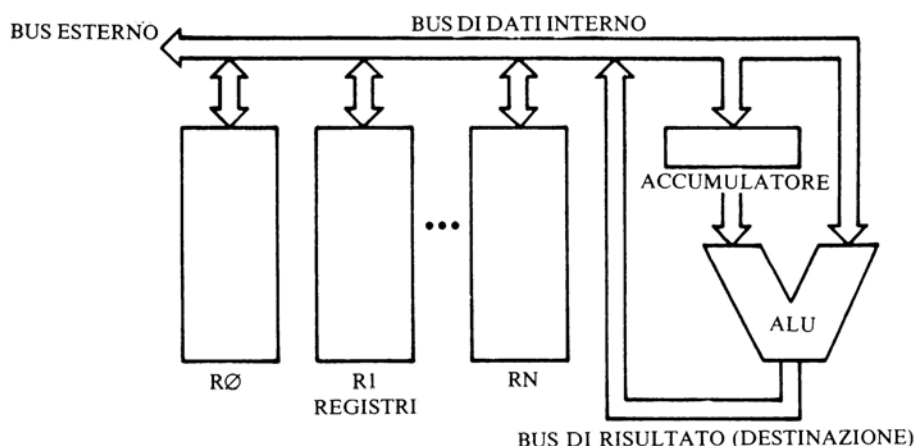



Fig. 4-2: Dentro il microprocssore.

effettuare realmente le operazioni aritmetiche e logiche specificate dall'istruzione. In aggiunta le ALU sono tipicamente equipaggiate con uno *shifter* (dispositivo di scorrimento) che fa scorrere il risultato di una o più posizioni di bit a sinistra o destra (per più dettagli vedere i riferimenti C201 o C202). Connesso alla ALU c'è un registro speciale chiamato *status o flag* register che contiene *flags* (bandierine). Un *flag* è un bit speciale che memorizza una condizione interna. Tali condizioni speciali sono un riporto aritmetico, un risultato uguale a zero, un risultato negativo, o altri eventi speciali. Istruzioni speciali dentro al programma possono verificare queste condizioni e causare branches (diramazioni del programma) in funzione di determinati eventi.

Eseguiamo una semplice addizione del contenuto di due registri così da illustrare il modo in cui la CPU opera. Supponiamo che i registri R0 ed R1 siano stati entrambi caricati per mezzo di istruzioni specifiche con dati di 8 bit. Ogni registro contiene 8 bit binari. Il nostro obiettivo è di sommare i contenuti di R0 ed R1 e depositare il risultato in R0. Alla fine dell'operazione il contenuto di R1 deve essere intatto, ed

R0 deve contenere la somma dei due numeri. L'istruzione di *addizione* farà esattamente questo. Il contenuto di R0 viene inviato verso la cima dell'illustrazione e verso l'ingresso destro della ALU. Quindi il contenuto di R1 viene inviato lungo lo stesso data bus interno verso l'ingresso sinistro della ALU.
Quindi viene trasmesso un ordine di addizione alla ALU della control unit. La ALU somma, e poi trasmette il risultato dell'addizione sulle sue linee di uscita verso il fondo dell'illustrazione, sulla destra. Questa somma viene mandata alla fine al registro di destinazione, R0 nel nostro esempio. *La lettura di un registro non cambia il suo contenuto.* Il contenuto di R1, quindi, non viene influenzato dall'operazione di addizione. Comunque, siccome la somma è stata scritta in R0, R0 contiene la somma dei due numeri alla fine dell'operazione; il suo contenuto originale è stato cancellato. Si può notare che, come risultato dell'addizione, saranno assegnati automaticamente dalla ALU uno o più flags (in particolare: riporto, zero, segno).

Questa era una tipica istruzione di addizione. Non servirebbe a nessuno scopo, qui, scendere in maggiori dettagli, ed il lettore interessato dovrebbe consultare uno dei nostri riferimenti per avere informazioni più dettagliate.

Riassumiamo qui il funzionamento di una CPU: la Central Unit va a prendere un'istruzione dalla memoria, la porta nell'instruction register (registro delle istruzioni), la decodifica e quindi genera i segnali di controllo appropriati automaticamente.

La ALU è incaricata di eseguire l'operazione specificata, sia sui registri interni del MPU, sia talvolta direttamente sui dati forniti dal mondo esterno attraverso il data bus (bus dei dati). Tali dati potrebbero essere forniti dalla *memoria* o da un dispositivo di *input-output.*
Ora è il momento di dare un'occhiata al mondo esterno: *la memoria e l'input-output.*

La memoria

Si è visto che la memoria immagazzina il *programma*. La memoria deve anche immagazzinare i *dati* che l'utente ha introdotto o che si creano durante l'esecuzione del programma. Si è messo in evidenza che le memorie MOS (memorie che usano circuiti integrati) hanno un inconveniente importante attualmente: le memorie a lettura/scrittura, cioè memorie dove l'informazione può essere sia scritta che immagazzinata all'interno (per una lettura successiva) sono *volatili. Il loro contenuto scompare quando si toglie l'alimentazione.* Per questa ragione, si usano due tipi di memoria nei sistemi dei microcomputer: le ROM e le RAM. Nei sistemi più antiquati come quelli dei minicomputer, questo problema non esisteva. La tecnologia usata per realizzare le memorie era quella dei nuclei. Ogni bit di informazione (cioè uno "0" logico o un "1") era immagazzinato in un nucleo di ferrite, analogo come

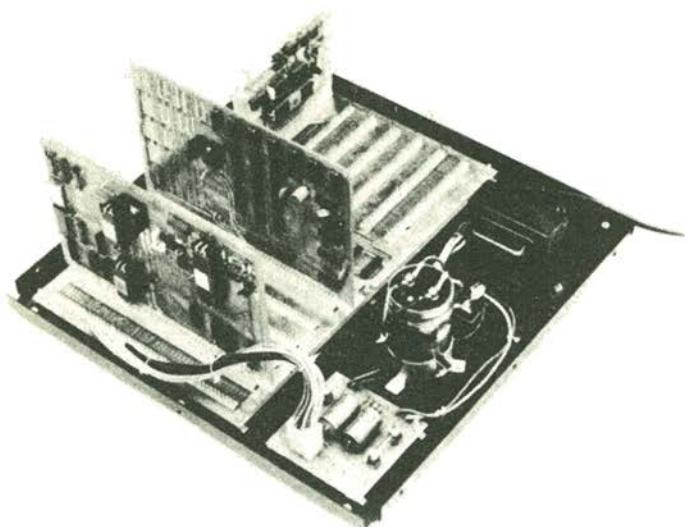



Fig. 4-3: Dentro il mobile del microcomputer (un SWTPC 6800): le tre piastre sulla sinistra sono la CPU, la memoria, l'interfaccia I/O. Sulla destra c'è l'alimentatore.

BUS DI DATI ESTERNO
BUS DI DATI INTERNO (8 BIT)
SP (16 BIT)
PC (16 BIT)
H L
REGISTRO Ø
...
REGISTRO N
ACCUMULATORI
FLAGS
C
O
N
H
P
ALU
CONTROLLO
SHIFTER
REGISTRI DI DATI A 8 BIT
BUS DI INDIRIZZAMENTO ESTERNO (16 BIT)

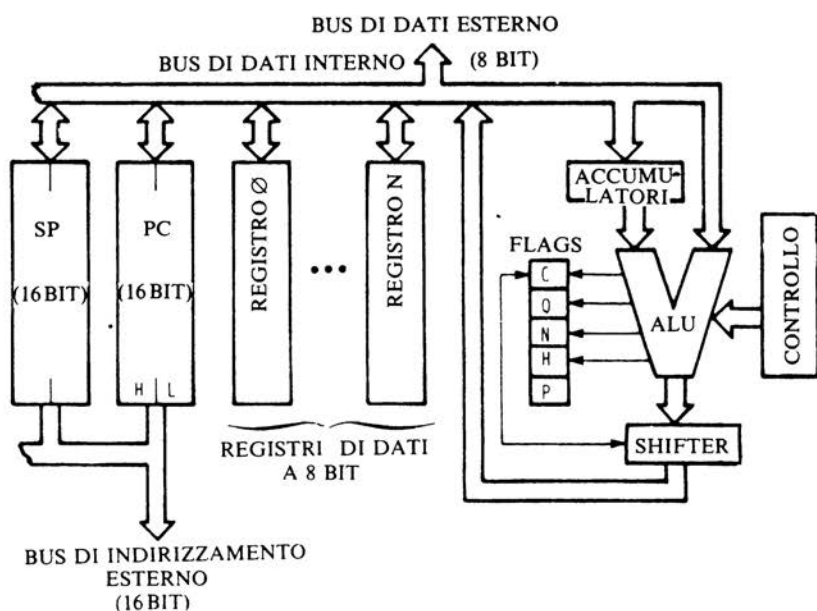



Fig. 4-4: Un programma più dettagliato mostra i flags, lo shifter e il PC.

forma ad una ciambella. Poichè il nucleo era sempre magnetizzato in una direzione o nell'altra, questa memoria non richiedeva l'alimentazione per mantenere l'informazione. In vista dell'utilizzazione della tecnologia MOS è ora diventato necessario usare *due* tipi di memoria dentro al sistema.

Il primo tipo di memoria è la RAM o *Random Access Memory* (memoria ad accesso casuale). La RAM *è una memoria che può essere sia letta che scritta.* Chiaramente tutte le memorie dovrebbero essere così se non fossero volatili. La RAM deve essere usata in un sistema per i *dati,* altrimenti non sarebbe possibile al MPU immagazzinare i risultati dei suoi calcoli, o immagazzinare i dati che gli provengono dalla tastiera o da un altro dispositivo di input. *La dimensione di un programma o file che voi volete caricare in un sistema a microcomputer è limitata dalla quantità di RAM che il sistema ha.*
Si vedrà nella prossima sezione che vengono usati due tipi di tecnologie per realizzare la RAM: memorie statiche e dinamiche.

Il secondo tipo è la ROM (o Real Only Memory - memoria a sola lettura). La ROM non è volatile, ed è usata per immagazzinare programmi. Una volta che l'informazione è stata depositata nella ROM, in linea di principio non può più essere cambiata. A condizione che il programma sia corretto, esso non verrà più cambiato, e può quindi risiedere nella ROM. Questo è il caso della maggior parte delle applicazioni industriali. In particolare il programma "monitor" (di controllo) del vostro microcomputer è normalmente nella ROM, poichè non c'è intenzione di cambiarlo. Comunque la maggior parte degli utenti che sviluppano un programma da sè vogliono avere la possibilità di cambiarlo. Per questa ragione, sono disponibili tipi alternativi di ROM chiamati PROM o RPROM. Benchè le abbreviazioni possono variare, una PROM (Programmable Read Only Memory) è una memoria a sola lettura programmabile dall'utente. Essa sfrutta la tecnologia del collegamento elettrico interno a fusibile, che permette all'utente di depositare degli 0 e degli 1 in essa, usando un *PROM programmer* (programmatore per PROM) di costo relativamente basso. Le PROM sono economiche. Comunque, siccome esse usano la tecnologia del collegamento a fusibile, una volta programmate (collegamento fuso) non possono più essere cambiate. Esse non sono ancora adatte per i nostri scopi. Il tipo più frequentemente usato è la RPROM o EPROM (*Reprogrammable* PROM o *Eraseable* PROM). Questo sta per PROM riprogrammabile o PROM *cancellabile con raggi ultravioletti,* o PROM *elettricamente cancellabile.* Con questo tipo di memoria l'utente può "programmare" il chip, eseguire il programma e più tardi cancellare il contenuto e riprogrammare. Questo tipo di memoria è usato normalmente in qualsiasi sviluppo di un programma fisso.

Comunque, le applicazioni di hobbisti e commerciali differiscono dalle applicazioni industriali. Nelle applicazioni hobbistiche e commerciali deve essere eseguita una varietà di programmi lungo tutta la giornata, programmi che devono risiedere *in successione* nella memoria del sistema. *Per questa ragione* le ROM, PROM o EPROM *non sono usate in tali sistemi* (ad eccezione che per il programma del costruttore). Le EPROM sono destinate a programmi fissi che risiedono perma-

nentemente nel sistema. Al massimo, questo sarà il caso del programma "monitor" nel programma del sistema hobbistico o commerciale. Comunque *i programmi dell'utente in tali sistemi risiedono sempre nelle RAM.*
L'utente deve essere consapevole del fatto che ogni volta che si toglie l'alimentazione il suo programma scompare. Questo è il motivo per cui ogni sistema di questo genere deve essere equipaggiato con una *mass memory* (memoria di massa) ausiliaria, come un floppy disk, sul quale il programma viene registrato permanentemente. Ogni volta che l'utente intende eseguire un programma, il programma deve essere riprodotto dal disk nel sistema. Questa è un'operazione semplice e veloce che può anche essere fatta automaticamente se il sistema ha un buon *disk operating system* (DOS). L'operazione quindi è invisibile per l'utente.

RAM statiche e dinamiche

I dettagli della tecnologia non dovrebbero interessare l'utente. Semplicemente, per sistemi destinati ad avere una piccola memoria viene di solito utilizzata la RAM statica poichè porta come risultato ad una scheda economica. In sistemi più grandi, diciamo più di 16K di memoria e quindi la maggior parte delle schede di memoria sono dinamiche. Purchè il prezzo sia giusto, si può usare una scheda di RAM statica senza alcun impatto di tipo tecnico sul sistema. Questo argomento non sarà discusso ulteriormente in questa sede.

Fig. 4-5: Un PROM programmer in formato valigetta. Gli zoccoli servono per inserire le PROM. Il "bit pattern" viene introdotto attraverso la tastiera esadecimale.

L'Input-Output

I dispositivi di input-output tipici in un sistema microcalcolatore consistono in una *tastiera* per l'input (ingresso), e un *display CRT* per l'output (uscita). Si possono usare molti altri dispositivi. Questi dispositivi di solito forniscono o richiedono dati di 8 bit in parallelo. In altre parole, una tastiera tipica mette su un bus 8 bit corrispondenti al codice del carattere che è stato premuto.
In modo analogo quando un carattere è inviato ad un display CRT, viene mandato in forma di 8 bit paralleli. Altri dispositivi di input- output preferiscono uno schema di trasmissione seriale (un bit dopo l'altro). Una *Teletype* è uno di essi. La *Teletype* è una comune *"telescrivente" con meccanismo di stampa ed una tastiera simile ad una macchina da scrivere. Quando si preme un tasto sulla Teletype viene inviata una successione di 11 bit verso il sistema di microcomputer. Il primo bit è chiamato il bit di "start" (inizio). È seguito da 8 bit di dati* che codificano il carattere ed il messaggio è terminato da due bit di stop. Come al solito 8 bit portano l'informazione che rappresenta il carattere al microcomputer. Comunque, questa volta essi sono trasmessi *serialmente* cioè uno dopo l'altro.

Riassumendo, ci sono dispositivi di input *seriali e paralleli.* Sono stati creati due tipi di chip d'interfaccia così da interfacciarli facilmente al bus dei dati del microcomputer. Dispositivi di input-output più complessi richiedono una *logica di interfaccia* più complessa in aggiunta a questi due chips che verranno descritti breve-

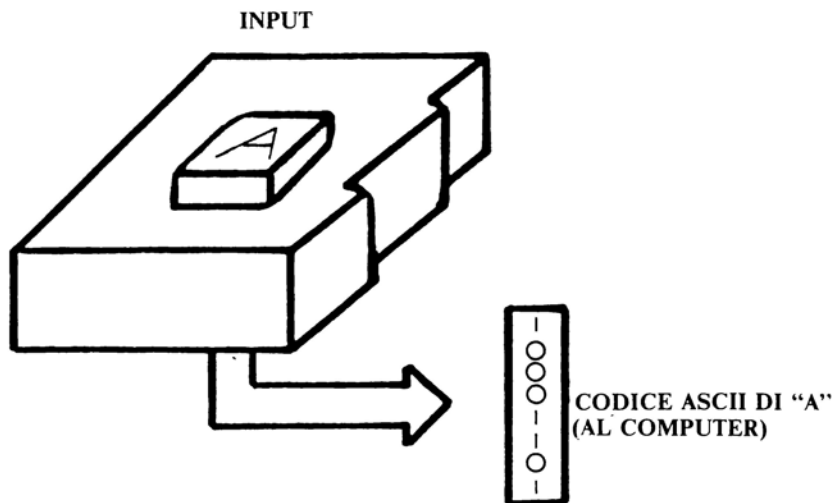



Fig. 4-6: Premendo un tasto sulla tastiera si genera un codice di 8 bit che viene trasmesso in parallelo.

mente. Essi richiederanno una *scheda d'interfaccia.* Inoltre è necessario che un codice di 8 bit mandato dal sistema del microcomputer al dispositivo sia da quest'ultimo interpretato come *comando* e non semplicemente come dato. In questo caso, bisogna fornire un *command decoder* (decodificatore di comandi): questa è una *scheda di controllo.* Una scheda di controllo riceverà il comando di 8 bit, lo decodificherà, e lo tradurrà in una sequenza di gradini che porteranno a termine il comando specificato. *Controllers* sono usualmente necessari per displays CRT intelligenti e per unità disk driver.

I due chip di base che vengono usati in un sistema microcalcolatore per fornire un'interfaccia seriale o parallela sono rispettivamente l'UART ed il PIO. Nel caso del PIO le abbreviazioni variano da costruttore a costruttore. Per semplicità, qui useremo l'abbreviazione PIO.

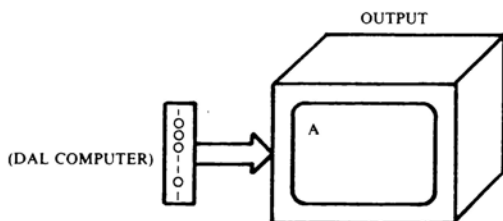



Fig. 4-7: Il codice binario di A viene mandato al display CRT.

UART sta per "Universal Asynohronous Receiver Trasmitter" (Ricevitore Trasmettitore Asincrono Universale) i dettagli del suo funzionamento non ci interessano più. Un diagramma dell'UART appare in fig. 4-8. *Il ruolo dell'UART è quello di convertire parallelo in seriale e seriale in parallelo.* Un UART accetta un ingresso parallelo di 8 bit, e lo converte in una sequenza di 8 (o più) bit su una linea seriale. Simultaneamente, esso può ricevere segnali seriali in ingresso e convertirli in una uscita parallela a 8 bit. L'ingresso parallelo a 8 bit, e l'uscita parallela a 8 bit sono di solito collegati ai data bus bidirezionali del microprocessore. L'ingresso seriale e l'uscita seriale dell'UART saranno collegati a dispositivi specifici.

PIO sta per "parallel programmable input output". Anch'esso naturalmente si collega al data bus del microprocessore, e crea due o più *porte* di I/O (I/O sta per "input/output"). Una *porta* è semplicemente un collegamento a 8 bit (in questo caso) col mondo esterno. Oltre che una singola connessione a 8 bit col data bus, essa crea due o più collegamenti col mondo esterno. In aggiunta esso fornisce un *buffering interno:* ogni porta I/O è equipaggiata con un registro a 8 bit *che memorizza l'informazione* (un "buffer"). In aggiunta, un PIO *è programmabile,* e può maneggiare *protocolli* di controllo I/O. Ciò si chiama *handshaking.*

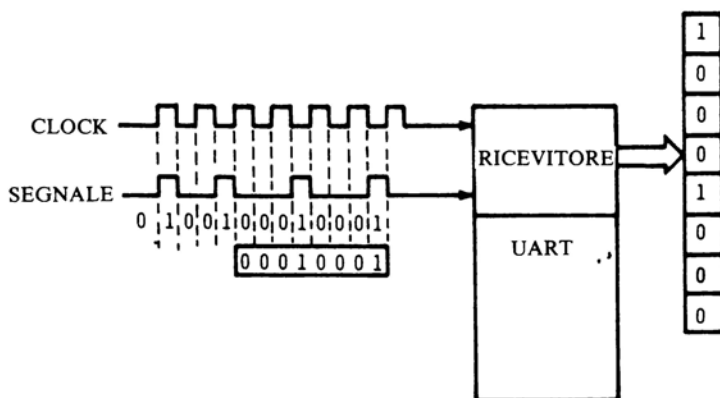



Fig. 4-8: L'UART converte la trasmissione seriale in parallela e quella parallela in seriale.

Qui è sufficiente dire che il PIO e l'UART sono chip universali trovabili sulla maggior parte delle schede dei microcomputer per fornire un'interfaccia comoda per dispositivi sia seriali che paralleli. Si può aver bisogno di altri circuiti logici se la funzione non è soltanto quella di trasmettere caratteri ma anche trasmettere comandi o informazioni di status (condizione in cui si trova un certo dispositivo): può essere necessario che l'informazione sia decodificata ed eseguita dalla sua scheda di controllo. In un caso simile, è necessaria anche una *scheda di controllo.*

Sono stati descritti tutti i chip basilari di un sistema microcalcolatore. Esistono anche chips addizionali per scopi speciali per facilitare altre funzioni. In particolare le funzioni di controllo di un dispositivo sono ora facilitate da controllori di dispositivi ad un chip quali il FDC o "floppy disk controller" o il CRTC o "CRT controller". Esistono anche convertitori analogico-digitali a chip singolo. Per una discussione dettagliata di questi componenti, il lettore è nuovamente rimandato a C201 e C207.

L'Alimentazione

L'alimentatore è incaricato di fornire ai circuiti una tensione (o tensioni) stabilizzata. È un elemento importante, sia in termini di costo che di affidabilità dell'intero sistema. È un fattore importante anche fisicamente, poichè spesso occupa un quinto del volume del contenitore in cui c'è il microcomputer.

I quattro elementi funzionali di un alimentatore appaiono in fig. 4-9.

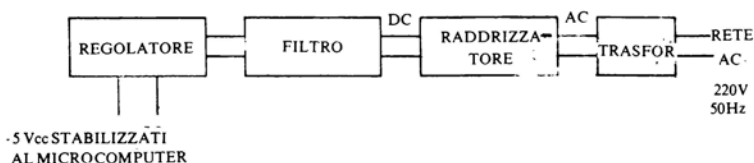



Fig. 4-9: I quattro elementi di un'alimentatore.

SOMMARIO

Sono stati descritti i tre elementi funzionali di un sistema microcalcolatore: la CPU, la memoria, l'input-output. Ognuna di queste funzioni può essere compiuta da chip LSI specializzati. Tutti questi chip possono essere montati su una o più schede, che costituiscono le schede del microcomputer. È stata descritta l'esecuzione di un'istruzione all'interno della CPU.
Ora siamo pronti ad usare il sistema. Usare il sistema significherà sia scrivere un programma sia immettere un programma nella memoria del sistema, il quale eseguirà le funzioni richieste. Esaminiamo ora cosa comporta il programmare.

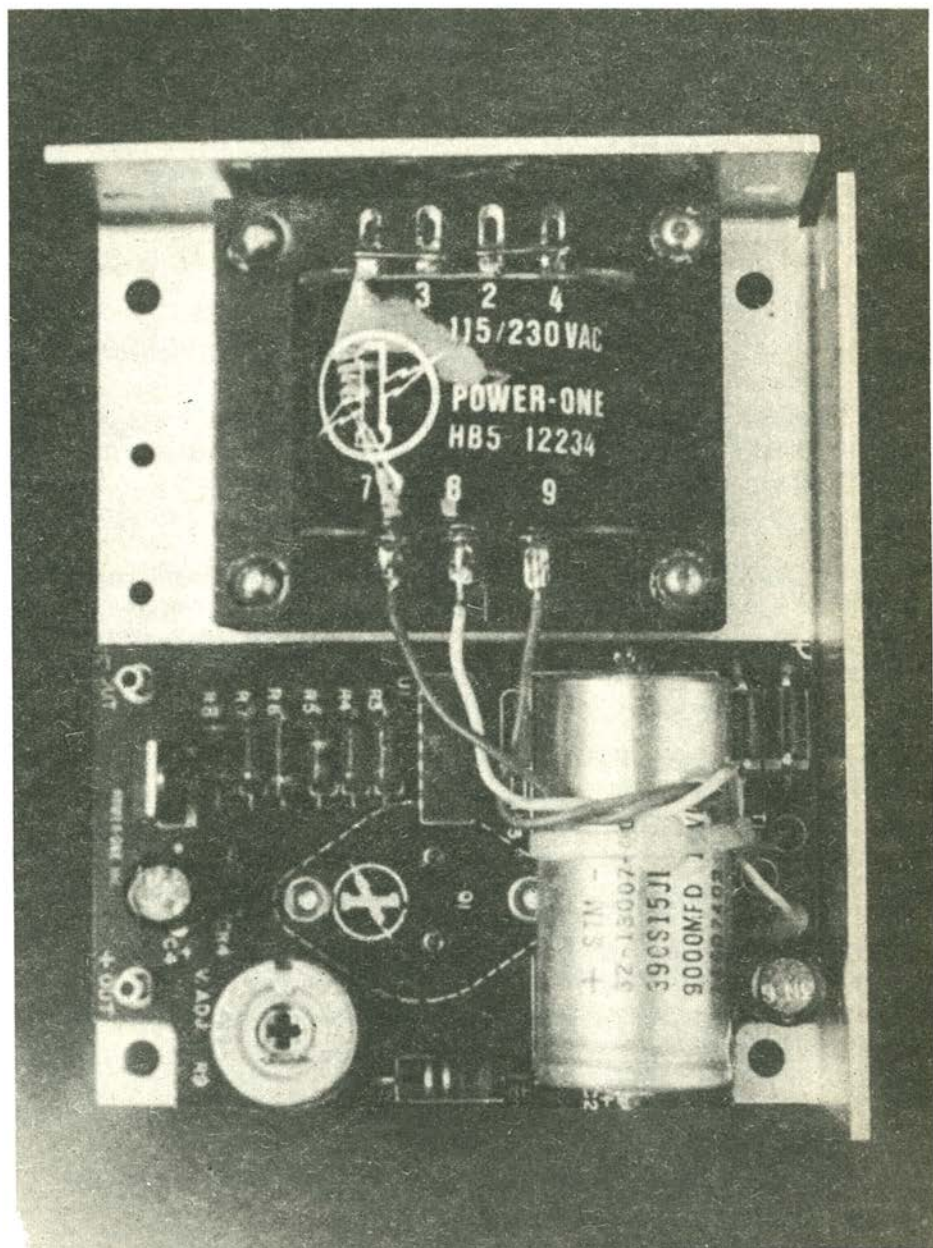



Fig. 4-10: Alimentatore di un microcomputer con i suoi elementi funzionali (della "Power one").

CAPITOLO 5

PROGRAMMAZIONE

DEFINIZIONI DI BASE

Un *programma* è una sequenza di ordini o *istruzioni,* dati a un computer, per risolvere un problema specifico. Nel campo delle applicazioni commerciali, questo è chiamato Electronic Data Processing (Elaborazione Elettronica dei Dati). In tale ambiente la funzione principale di un programma è di trattare files che contengano dati.

Nel caso di personal computing i programmi possono essere molto vari. Essi possono fare giochi, controllare un sistema d'allarme dentro la casa, o fornire possibilità di word processing.

Idealmente, sarebbe desiderabile dire cosa fare ad un computer in semplice inglese. Sfortunatamente, è stato provato che il caso detto "linguaggio naturale" (inglese o altre lingue) non può essere usato per comunicare con un computer, poichè è troppo ambiguo. Si può usare per specificare ordini solo un piccolo sotto-insieme del linguaggio naturale. Per di più, la necessità di una elaborazione efficiente richiede che le istruzioni siano di una lunghezza ben definita. Per questa ragione, sono stati creati *linguaggi di programmazione,* che sono strumenti efficienti (dal punto di vista del computer) per eseguire programmi. La loro efficienza *a livello dell'utente* sarà valutata più avanti in questo capitolo. Infine l'unico "linguaggio" che qualsiasi computer comprende realmente, consiste di una sequenza di 0 e 1, cioè istruzioni *espresse nel sistema binario.* Questo è chiamato *linguaggio macchina.*

Una volta che il problema, è stato definito, la sua soluzione sarà specificata come *algoritmo.* L'algoritmo è semplicemente una specificazione passo per passo della soluzione del problema. Come esempio, ecco un algoritmo semplificato per il controllo del traffico ad un incrocio:

- Accendi il verde per la direzione A.
- Aspetta due minuti.
- Spegni il verde.
- Accendi il giallo per la direzione A.
- Aspetta trenta secondi.

Spegni il giallo.
- Accendi il rosso.
- Accendi il verde per la direzione B.
- Aspetta un minuto.
- Accendi il giallo per la direzione B.
- Aspetta venti secondi.
- Accendi il rosso per la direzione B.
- Ritorno al primo step (passo).

L'*algoritmo* di cui sopra si chiama "fixed cycle loop" (loop a ciclo fisso) per un incrocio tipico. Naturalmente un microcomputer può essere usato per favorire, una risposta più dinamica all'incrocio. *L'algoritmo può essere migliorato.* L'esempio sopra fornisce l'esempio di un *loop.* Si può vedere che dopo che è stato eseguito l'ultimo step viene eseguito di nuovo il primo. Il controllo in un loop va dall'ultima alla prima istruzione.

Programmare significa tradurre un algoritmo nel linguaggio del computer. La programmazione può essere fatta essenzialmente in due modi:

1 - In linguaggio a livello macchina

Per programmare in linguaggio a livello macchina occorre specificare istruzioni che il computer può eseguire immediatamente e prontamente. Poichè programmare direttamente in binario (usando degli 0 e degli 1) è tedioso e può portare ad errori, viene usata una rappresentazione simbolica delle istruzioni. Questo è chiamato *assembly-level language* (linguaggio a livello di assemblaggio). Per esempio: ADD RO, R1 significa "addiziona il contenuto di R0 ad R1 e deposita il risultato in R0" (in questo esempio). Questa istruzione è espressa in *forma simbolica.* Perché possa essere eseguita dal microcomputer, deve essere tradotta in bit binari. Questa traduzione viene realizzata da un programma speciale, *l'assembler.*

L'assembler è semplicemente un programma di traduzione automatica che accetta un assembly-level program, e lo traduce in codice macchina o codice binario per il microprocessore. Esso traduce ogni istruzione simbolica in un'istruzione binaria. Queste istruzioni binarie (lunghe ciascuna tipicamente 8, 16 o 24 bit) possono essere poste, quindi, nella memoria del sistema ed eseguite.

L'introduzione di un programma nella memoria del sistema può essere fatta direttamente, nel caso di una ROM (inserendo un componente) o attraverso una periferica. In un sistema tipico, il programma viene battuto sulla tastiera, trasferito nella memoria RAM, quindi immagazzinato in un memoria di massa come il disk. Per usare il programma, esso viene caricato dal disk nella memoria RAM del sistema dove può essere eseguito o tradotto.

2 - In linguaggio ad alto-livello

La seconda possibilità è di scrivere un programma di linguaggio a più alto livello. Un linguaggio ad alto livello è un linguaggio più vicino all'inglese parlato, ed è mol-

to più facile per l'utente programmare con esso. Frasi o istruzioni in linguaggio ad alto livello sono di solito abbastanza facili da capire ma obbediscono ancora ad una sintassi rigida in modo da eliminare tutte le ambiguità. C'è un grande numero di linguaggi ad alto livello disponibile. Comunque per scopi commerciali e personali, c'è solo un linguaggio ad alto-livello principale usato universalmente per i microprocessori al giorno d'oggi: questo è il linguaggio chiamato BASIC. La programmazione in BASIC sarà descritta in un capitolo successivo. Un altro linguaggio è molto usato in applicazioni di controllo ed industriali dei microprocessori. Questo è il PL/M (Programming language for Microprocessors - Linguaggio di Programmazione dei Microprocessori) inizialmente sviluppato dalla INTEL (e col trade mark della INTEL). Una delle differenze importanti tra il PL/M ed il BASIC è che il PL/M è un *compiler* (compilatore) mentre il BASIC è un *interpreter* (interprete). Chiariamo ciò.

Un *interpreter* è il programma speciale richiesto per interpretare ed eseguire il linguaggio ad alto livello come il BASIC. Naturalmente le istruzioni in linguaggio BASIC devono alla fine essere convertite in formato binario che è l'unico linguaggio che il computer può eseguire. Sono a disposizione due alternative. O ogni frase può essere tradotta in codice macchina in sequenza creando un'*object code* (codice oggetto che può essere eseguito separatamente) o altrimenti ogni riga del linguaggio ad alto livello può essere tradotta e quindi immediatamente eseguita prima di considerare la riga seguente. *Quando ogni riga viene immediatamente tradotta ed eseguita, si tratta di un programma interprete. Quando l'intero programma viene tradotto in object code, pronto per l'esecuzione successiva, si tratta di un compilatore.*
Il vantaggio di un interpreter è che ogni volta che una riga viene battuta sulla tastiera, essa può essere eseguita senza ritardo. Le righe possono essere cambiate liberamente nel programma senza incorrere in alcun ritardo per l'esecuzione. Questa è una grossa utilità per programmare efficientemente. Lo svantaggio dell'interpreter è che la maggior parte dei programmi contengono un grande numero di loop. Ogni volta che un loop viene eseguito ripetutamente, ogni frase viene tradotta di nuovo per essere usata. Quando si usa un compilatore, ogni riga è tradotta solo una volta e quindi viene realizzata l'esecuzione con alta efficienza sulla rappresentazione del codice macchina. Perché allora l'interprete è così popolare? Diamo un'occhiata al compilatore.

Un compilatore traduce l'intero programma in codice macchina. Quindi solo dopo un ritardo considerevole dovuto alla traduzione del programma completo, si può procedere all'esecuzione. Se poi accade di dover cambiare una riga nel programma, normalmente si deve ricompilare l'intero programma il che comporta un ritardo consistente. Per questa ragione gli altri interpreti sono più popolari per lo sviluppo di programmi poiché essi permettono *l'interazione diretta* entro il sistema. Si dice che gli interpreti sono interattivi; una istruzione può essere eseguita, cambiata, e quindi eseguita di nuovo. Ciò è molto soddisfacente. Per rimediare al rendimento più basso degli interpreti, sono disponibili anche *versioni* di BASIC *compila-*

tore. Una volta che un programma è stato completamente debugged (corretto) ed è pronto per essere eseguito senza ulteriori cambiamenti, un compilatore BASIC traduce il programma originale nella sua rappresentazione in codice macchina, che sarà poi eseguita.

È importante notare qui che molti utenti ottengono una prestazione sufficientemente veloce anche con un interprete e che i vantaggi di uno sviluppo del programma veloce e comodo compensano ampiamente la perdita in inefficienza intrinseca dell'interprete. Questo potrebbe non essere vero in un ambiente di controllo: e la maggior parte degli utenti in un ambiente di controllo possono prendere in considerazione un compilatore come il PL/M, una codificazione diretta in linguaggio a livello-assembley.

LINGUAGGI DI PROGRAMMAZIONE

È stato inventato un gran numero di linguaggi di programmazione. Ognuno di essi offre vantaggi specifici ad alcune categorie di utenti. Bisogna ricordare che il criterio principale per valutare un linguaggio di programmazione ad alto livello è di solito la convenienza per una specifica classe di utenti. Convenienza significa, da una parte, la facilità di programmazione nel linguaggio specifico per l'utente particolare, dall'altra, la disponibilità di molti programmi scritti in questo linguaggio che possono essere prontamente usati.

Poiché il PL/M è stato il primo compiler sviluppato per microprocessori, esso viene largamente accettato.

Ora cominciano ad essere disponibili diverse versioni del FORTRAN. Il FORTRAN è un compilatore vecchio e rispettabile il cui nome ne indica l'orientamento: "FORmula TRANslator". È ideale per calcoli scientifici ed esistono vaste biblioteche di programmi scritti in tale linguaggio.

Per ogni scopo pratico, il solo *interprete* molto usato attualmente è il BASIC. Un altro interprete sarebbe molto utile per il linguaggio APL, ma non è ancora pronto per i microprocessori.

ESECUZIONE DEI DIAGRAMMI DI FLUSSO

Preparare il diagramma di flusso (flow-chart) è fondamentale per programmare con successo. Un diagramma di flusso è semplicemente una rappresentazione simbolica dell'algoritmo che deve essere eseguito.

Vengono usati due simboli di base, con alcune variazioni: rettangoli e rombi.

Il *rettangolo* viene impiegato per ordini o "statements" (istruzioni) come "apri la porta" o "A = 2".

Il *rombo* viene usato per un test come "se bolle, allora aggiungi il C chimico" o se "A = 2, allora vai al punto 51 del diagramma di flusso" o se "X = 1, stampa ERRORE".

Consideriamo un esempio. Il seguente è un "algoritmo di un termostato".

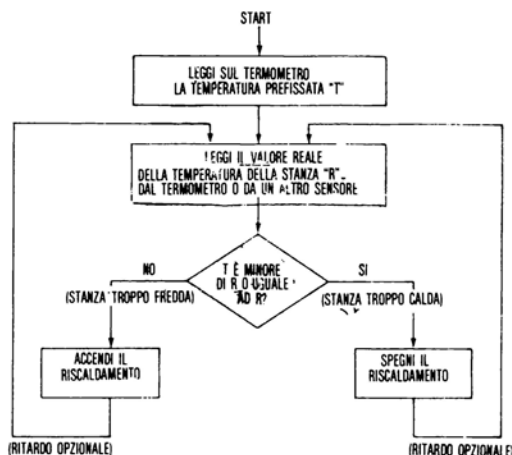



Fig. 5-1: Il diagramma di flusso rappresenta l'algoritmo.

Nello step "1", viene letta la temperatura stabilita manualmente sulla scatola del termostato: questo è "T".

Nello step "2", viene misurata con un opportuno dispositivo l'attuale temperatura della stanza "R".

- Se la stanza è troppo calda (R>T) (si legge R maggiore di T), allora il riscaldamento viene spento.
- Se la stanza è troppo fredda, viene acceso.
- Questo è ciò che avviene negli step "3", "4", "5".

Il simbolo a forma di rombo viene usato nella fase "3" poiché si ha un test. Dal rombo escono due frecce, considerando che il test può avere due esiti possibili. Ce ne potrebbero essere più di due.

DOMANDA *1: Se diverse frecce entrano in un simbolo, esso è necessariamente un rombo?*

DOMANDA *2: Ci può essere più di una freccia che parte da un rettangolo?*

RISPOSTE *1: NO; 2: NO*

Una volta che di un algoritmo sia stato fatto il diagramma di flusso, diventa una cosa molto più semplice programmarlo. Questo è un metodo caldamente raccomandato per scrivere programmi che funzionino:

1-sviluppare l'algoritmo corretto che specifica la soluzione del problema
2-preparare il diagramma di flusso
3-programmare nel linguaggio di vostra scelta.

L'esperienza indica che approssimativamente il 10% degli operatori può programmare direttamente dall'algoritmo nel linguaggio di programmazione, ed avere un programma corretto.
Sfortunatamente l'esperienza indica pure che il 90% dei soggetti crede di appartenere a questo 10%! In breve, la maggior parte delle persone inizialmente non crede alla necessità di fare diagrammi di flusso fino a che il loro programma non funziona. *Le probabilità sono contro di voi in maniera schiacciante se non fate il diagramma di flusso.*

Comunque l'esperienza indica pure che il 90% non crede a ciò fino a che il loro programma non fallisce. Perciò non discutiamone oltre.
È importante notare che fare il flow-chart è facile: non ci sono convenzioni o requisiti per quello che c'è scritto in una "scatola". Voi potete usare la lingua italiana, le vostre abbreviazioni, o persino un linguaggio di programmazione.

Il disegno illustra il "flusso" di esecuzione nel tempo, di qui il suo nome "diagramma di flusso".
Talvolta vengono usati simboli speciali per rendere più chiaro il diagramma di flusso, per esempio: simboli speciali per schede perforate, un disk, un nastro perforato.

RAPPRESENTAZIONE DELLE INFORMAZIONI IN UN MICROCOMPUTER

I circuiti di commutazione elettronica sono caratterizzati o dallo stato "0" o dallo stato "1". *Tutti i computers elettronici devono quindi rappresentare l'informazione immagazzinata nel sistema in forma binaria, cioé in forma di "0" e "1".* Il sistema binario rappresenta tutti i numeri in forma di sequenze di zeri ed uni. Per tutti gli scopi pratici un microprocessore a 8 bit immagazzina e manipola informazioni che sono lunghe 8 bit (una informazione di 8 bit si chiama "byte"). *Tutti i dati e le istruzioni* normalmente sono codificati in forma di uno o più byte.
Le istruzioni tipiche di un microprocessore sono lunghe uno, due o tre bytes. I caratteri sono codificati universalmente sotto forma di un byte, 8 bit consentono 2^8 combinazioni = 256 codici differenti = 256 caratteri differenti (solo 128 se viene usata la "parity", che usa un bit). La tabella ASCII, che appare in fig. 5-2, ci dà un esempio della rappresentazione dei caratteri.

La rappresentazione di grandi numeri interi richiede due o più bytes. Due bytes cioè 16 bit consentono solo fino a 64K = 65.536 combinazioni (1K = 1024). Questo è an-

NUMERI DI BIT															
								0	0	0	0	1	1	1	1
								0	0	1	1	0	0	1	1
								0	1	0	1	0	1	0	1
b_7	b_6	b_5	b_4	b_3	b_2	b_1	HEX 1 / HEX Ø	0	1	2	3	4	5	6	7
			0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
			0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
			0	0	1	0	2	STX	DC2	"	2	B	R	b	r
			0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
			0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
			0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
			0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
			0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
			1	0	0	0	8	BS	CAN	(	8	H	X	h	x
			1	0	0	1	9	HT	EM	)	9	I	Y	i	y
			1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
			1	0	1	1	11	VT	ESC	+	;	K	[	k	{
			1	1	0	0	12	FF	FS	,	<	L	\	l	¦
			1	1	0	1	13	CR	GS	–	=	M	]	m	}
			1	1	1	0	14	SO	RS	.	>	N	^	n	~
			1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Fig. 5-2: Tabella ASCII

ACK	Aknowledge (Riconoscimento)
BEL	Bell (Campanello)
BS	Backspace (Spazio)
CAN	Cancel (Cancelletto)
CR	Carriage return (Ritorno carrello)
DC1	Direct control 1 (Controllo diretto 1)
DC2	Direct control 2 (Controllo diretto 2)
DC3	Direct control 3 (Controllo diretto 3)
DC4	Direct control 4 (Controllo diretto 4)
DEL	Delete (Cancella)
DLE	Data link escape (Interruzione collegamento dati)
EM	End of medium (Elemento terminato)
ENQ	Enquiry (Domanda)
EOT	End of transmission (Fine trasmissione)
ESC	Escape (Interruzione)
ETB	End transmission block (Fine blocco trasmissione)

ETX	End text (Fine testo)
FF	Form feed (Alimenta)
FS	Form separator (Genera separatore)
GS	Group separator (Separatore di gruppo)
HT	Horizontal tab (Tabella orizzontale)
LF	Line feed (Alimetazione di riga)
NAK	Negative acknowledge (Riconoscimento negativo)
NUL	Null (Nullo)
RS	Record separator (Separatore di disco)
SI	Shift in (Scorri in)
SO	Shift out (Scorri out)
SOH	Start of heading (Inizio)
SP	Space (Spazio)
STX	Start text (Inizio testo)
SUB	Substitute (Sostituto)
SYN	Synchronous idle (Sincronismo di mantenimento)
US	Unit separator (Unità separatrice)
VT	Vertical tab (Tabella verticale)

Fig. 5-3: Abbreviazioni della tabella di codice ASCII.

cora insufficiente nella maggior parte dei casi, così che devono essere usati 3 o più bytes, per i numeri interi. Convenzioni speciali vengono usate per i numeri decimali: ciò viene chiamato "floating-point representation" (rappresentazione del punto fluttuante o rappresentazione a virgola mobile). Come esempio, di seguito appare la rappresentazione binaria dei numeri interi da 0 a 7. Per rappresentare otto combinazioni possibili sono necessari tre bit.

A proposito, abbiamo appena scoperto che le cifre da 0 a 7 possono essere usate per rappresentare qualsiasi combinazione di 3 bit binari. Questa è la rappresentazione *ottale*.

INTERO	RAPPRESENTAZIONE BINARIA
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

L'ottale è stato usato ampiamente con i microcomputer. È relativamente inefficiente, e può essere considerato una cosa del passato, a dispetto dell'insistenza di alcuni costruttori a fornire Octal (ottale) invece di hexadecimal (Esadecimale). Comunque questo è un punto poco importante.

OPERAZIONI LOGICHE

L'aritmetica binaria coi numeri binari è abbastanza semplice, almeno nel caso di numeri interi:

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 10$ (o 0, col riporto di 1)

Le quattro operazioni principali sui numeri binari sono: OR, AND XOR, NOT. L'OR è rappresentato da un simbolo a V e l'AND è rappresentato da un simbolo V capovolto Λ.

Ogni operazione può essere definita dalla sua "*tabella della verità*". Una tabella della verità è semplicemente il valore del risultato in funzione di ognuna delle variabili. Sotto appaiono quattro tabelle della verità per l'OR, l'AND, lo XOR, il NOT:

A	Ā
0	1
1	0

A	B	A Λ B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A V B
0	0	0
0	1	1
1	0	1
1	1	1

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

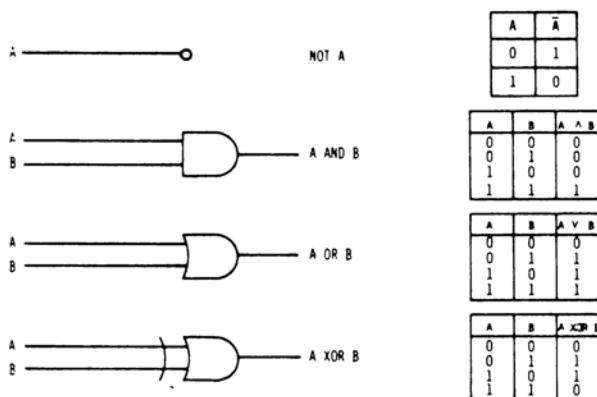



Fig. 5-4: Le tabelle della verità di base.

RAPPRESENTAZIONE ESTERNA

L'informazione può essere rappresentata esternamente in qualsiasi modo usale. Tipicamente, viene rappresentata *simbolicamente*, con simboli alfanumerici, e nu-

meri decimali. Ogni volta che non si hanno gli strumenti idonei per convertire il numero binario nella sua rappresentazione alfanumerica, si usa l'*esadecimale.*

La rappresentazione *esadecimale* è semplicemente un modo per codificare quattro bit binari (un "nibble") in un simbolo solo. Quattro bit possono generare fino a 16 combinazioni (hex significa 6 in greco). Vengono usati quindi i dieci simboli da 0 a 9, e i sei simboli successivi sono semplicemente le lettere da A ad F. In questo modo un byte ad 8 bit può essere rappresentato da due simboli esadecimali. In fig. 5-5 appare la tavola di conversione da binario ad esadecimale.

DECIMALE	BINARIO	ESADECIMALE	OTTALE
0	0000	0	00
1	0001	1	01
2	0010	2	02
3	0011	3	03
4	0100	4	04
5	0101	5	05
6	0110	6	06
7	0111	7	07
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

Fig. 5-5: Tabella di conversione esadecimale-ottale.

Per esempio, FF in esadecimale sta per "1111 1111" in binario. L'esadecimale è conveniente, poichè è molto più facile da memorizzare e usare del binario. In aggiunta, l'esadecimale può essere visualizzato comodamente ed economicamente con diodi emettenti luce (LED) o immesso in un sistema attraverso una tastiera a 16 tasti. Sulle schede dei microcomputers più economici tutto l'input (ingresso) è esadecimale ed anche l'output (usato) è esadecimale.

ESERCIZIO: *Qual'è la rappresentazione esadecimale di "1010 1010"?*

RISPOSTA: "A A".

SVILUPPARE UN PROGRAMMA

Il primo passo è di disegnare un diagramma di flusso. Poi il programma viene scritto sul foglio nel linguaggio scelto sia in linguaggio a livello assembly sia in linguaggio ad alto livello. Il programma viene inserito nella memoria principale del sistema (la RAM) attraverso una tastiera. Verranno quasi sempre fatti degli errori nella fase d'inserimento di questo programma. Per questa ragione, non è solo conveniente ma di fatto indispensabile avere un modo comodo per introdurre correzioni, spostare il testo, eseguire inserzioni, cancellature, o cercare un modello specifico.

Questa è la funzione dell'*editor* program. L'*editor* è un programma standard disponibile su qualsiasi sistema, è progettato per facilitare l'introduzione di dati o programmi. Poiché l'editor non sarà mai cambiato dall'utente, esso risiede normalmente nella ROM. In taluni sistemi, per liberare spazio nella memoria, accade che l'editor debba risiedere in una memoria esterna come un disk file, e venga caricato nella memoria centrale del sistema prima del suo uso.

Dopo essere stato battuto sulla tastiera, e dopo che gli errori di battitura sono stati corretti, il programma risiede nella memoria centrale del sistema, in forma simbolica. A questo punto il programma è una stringa (una sequenza di caratteri e numeri) di solito codificati nel formato ASCII descritto prima. Supponendo che il programma sia stato scritto in linguaggio a livello assembly esso deve essere tradotto dalla sua rappresentazione simbolica nel codice oggetto che è l'unico che il microcomputer possa eseguire. Il programma *assembler* viene ora caricato da un disk file nella memoria centrale del sistema (la RAM). L'utente chiede all'*assembler* di tradurre il programma *simbolico* in *codice oggetto.* L'assembler realizza questa traduzione automaticamente e fa una diagnosi, se rileva degli errori di sintassi. Supponendo che l'assembly sia stato corretto (non siano stati trovati errori di sintassi dall'assembler) il codice object è ora pronto per essere eseguito. Spesso si deve caricare il codice object in un zona della memoria differente da quella che si intendeva originariamente, così che è necessario un programma *loader* (caricatore) per collocarlo nella locazione di memoria adeguata, e così da correggere automaticamente tutti gli address (indirizzi di memoria) che il programma potrebbe contenere. Supponiamo ora che il codice object risieda nella memoria del sistema e sia pronto per essere eseguito. Poiché la prima volta la maggior parte dei programmi difficilmente viene eseguita correttamente, se noi andassimo avanti ed eseguissimo questo codice object, probabilmente l'esecuzione non sarebbe corretta. Sfortunatamente, in pratica, non ci sarebbe nessun modo di rilevare cosa è andato male, poiché non accadrebbe nulla di visibile.

Per determinare cosa non va, e dove, è necessario un programma *debugger* (correttore). Debugging vuol dire rintracciare e rimuovere gli errori da un programma. Un debugger consente di fermare automaticamente l'esecuzione di un programma a locazioni selezionate (break points), esaminare il contenuto del registro e della

memoria, o modificarli. Quindi prima di far girare il programma l'utente deve fissare alcuni break points: essi sono semplicemente addresses (indirizzi di memoria) arrivato ai quali il debugger ferma automaticamente l'esecuzione del programma. Il programma viene poi eseguito sotto il controllo del debugger. Ogni volta che viene raggiunto uno dei break points, il debugger ferma l'esecuzione. L'utente può esaminare il registro e la memoria ed assicurarsi che i risultati siano corretti. Se i risultati non sono corretti, allora il segmento di programma che ha causato l'errore può essere isolato. Questa diagnosi degli errori può essere affinata fissando break point addizionali nel modo necessario. Di solito vengono trovati molti errori e l'utente deve inserire correzioni. L'intera procedura appena descritta deve essere eseguita di nuovo. Ogni volta che il programma è stato "debugged" completamente e funziona con successo, esso è "corretto" (o piuttosto si presume sia corretto). In pratica, più tardi vengono quasi sempre trovati degli errori, in un programma vasto. Comunque è abbastanza corretto da funzionare per lo scopo a cui è destinato.

Il programmatore inesperto potrebbe pensare sia che il suo programma è totalmente corretto o altrimenti preoccuparsi del fatto che il suo programma non sarà mai corretto. Per qualsiasi programma lungo è probabilmente vero l'ultimo caso. È semplicemente impossibile in pratica evitare tutte le possibili combinazioni o situazioni che potrebbero innescare un comportamento strano in un programma. Comunque questo non significa che il programma non dia affidamento. Nella maggior parte dei casi il programma verrà eseguito correttamente, praticamente sempre. Solo se accadono situazioni molto strane ed inconsuete, come quella che si ha quando un altro utente usa questo programma per altri scopi, possono venire alla luce errori nascosti. Di solito questi errori non sono gravi e non comportano un mal funzionamento (essi sono chiamati "soft errors" errori lievi). Comunque questa è la ragione per cui sistemi altamente complessi possono crollare diverse volte in un giorno senza alcuna ragione apparente. La complessità di tali sistemi è semplicemente così alta che le probabilità di bugs (errori) nascosti che possano innescare il fallimento di un sistema sono alte. La maggior parte dei programmi degli utenti non raggiungeranno mai questo livello di complessità. Quindi li si può giudicare corretti per tutti gli scopi pratici. Essi saranno molto più corretti ed affidabili della maggior parte dei dispositivi meccanici complessi, come per esempio una automobile nuova.

SOFTWARE RICHIESTO PER LO SVILUPPO DEL PROGRAMMA

Si è visto che il minimo software richiesto per lo sviluppo di un programma consiste di un *editor,* un *assembler,* un *interpreter* (interprete), o un *compiler* (compilatore) *se viene usato linguaggio ad alto livello, ed un debugger.* Ogni sistema discreto fornisce questi mezzi. Deve essere inoltre disponibile un *disk operating system* (DO) (sistema di gestione dei disk) se si intende usare la memoria a disk. Un DOS è il programma (fornito dal costruttore) che automatizza l'uso dei disk. La potenza dei disk operating system varia notevolmente, a seconda del costruttore. Idealmente,

usando un DOS, dovrebbe essere possibile manipolare i files simbolicamente senza doversi preoccupare della loro reale collocazione fisica. Il sistema dovrebbe prendersi cura di tutto quello che si deve fare ed eseguire automaticamente i trasferimenti. Ciò sarà discusso nella sezione dei disk del capitolo sulle periferiche.

Se si ha intenzione di usare un linguaggio interattivo come il BASIC, allora dovrebbe essere disponibile un "*interprete*" per quel linguaggio. Infine devono essere disponibili facilities addizionali, quali un "*simulatore*" o un "*emulatore*". Qui non si parlerà di questi argomenti poiché essi esulano dallo scopo di questo libro (vedi C201 per esempio).

Infine, dobbiamo menzionare il programma monitor. Ogni computer viene equipaggiato con monitor residente nella ROM, o altrimenti, non potrebbe essere usato: non ci sarebbe alcun modo di battere un comando sulla tastiera, poiché il microprocessore non fa nulla fino a che non inizia ad eseguire un programma. Ogni volta che il sistema viene avviato, viene attuato un "Reset". Poi il microprocessore inizia ad eseguire un programma che risiede in una locazione di memoria specifica (diciamo address 0, per esempio). Questo programma può clear (cancellare) i registri, perfino verificare che ogni cosa stia funzionando appropriatamente, facendo girare alcune diagnostiche, poi incomincerà il *monitoring* della tastiera, e non farà nient'altro.

L'utente alla fine premerà un tasto sulla tastiera, ed il programma monitor riceverà il carattere e lo elaborerà.

Il "monitor" spesso include un certo numero di strumenti di debugging quali conversione esadecimale-ottale-binario e facilities per esaminare/cambiare registri o memorie.

ALLORA VOLETE PROGRAMMARE?

Avete due possibilità:

Se volete usare il microcomputer, e non siete interessati a cosa accade all'interno, non c'è alcuna ragione per usare istruzioni a livello assembly, che manipolano registri. Userete un linguaggio ad alto livello, come il BASIC. Per imparare come si fa, leggete il prossimo capitolo.

La seconda possibilità è di programmare in linguaggio assembly o persino in esadecimale. Se siete soddisfatti dell'esadecimale, ed avete un budget preventivo limitato, tutto ciò di cui avete bisogno è una scheda a microcomputer più un'alimentazione. La scheda dovrebbe avere una tastiera, per introdurre istruzioni o dati, ed un display a LED/ preferibilmente con sei LED, affinché sia l'address (indirizzo) che i dati a quell'address, possano essere visualizzati simultaneamente (in esadecimale). In aggiunta, è di grande convenienza avere un'interfaccia a registratore a cassette direttamente sul board così che i programmi possono essere salvati per un uso successivo, piuttosto che battuti di nuovo sulla tastiera.

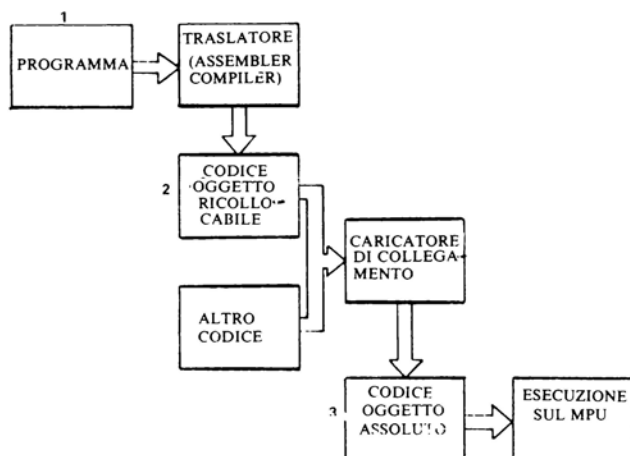



Fig. 5-6: Una sequenza di sviluppo di un programma.

Ma un personal computer minimo, con una tastiera completa, un assembler, BASIC e display CRT può essere ora acquistato per circa il doppio del costo di un board di microcomputer. Per tutto il resto di questo libro, quest'ultimo è lo strumento che useremo.

CAPITOLO 6

BASIC e APL

Il BASIC è un linguaggio di programmazione "ad alto livello". È un acronimo per "Beginner's All-purpose Symbolic Instruction Code" (Codice di Istruzioni Simbolico per tutti gli scopi per Principianti) e fu sviluppato al Dartmouth College.

Il BASIC, essenzialmente una versione ridotta del prestigioso linguaggio FORTRAN, viene facilmente imparato dai non specialisti nell'arco di qualche ora.

Inoltre, il BASIC viene eseguito da un "interprete": ogni istruzione può essere eseguita appena la si batte sulla tastiera (si dice che il linguaggio è "interattivo"). Il computer scopre ed indica immediatamente un errore di sintassi, permettendo una correzione immediata.

Guardiamo alcuni esempi pratici ed osserviamo le regole di programmazione.

```
10 PRINT "THE SUM OF 2+3 IS" 2 + 3 (Stampa "la somma di 2+3 è" 2+3
20 END                             (Fine)
```

Ogni riga è un'istruzione. L'esecuzione di questo semplice programma è ottenuta battendo "RUN" (svolgi il programma) e risulta:

THE SUM OF 2 + 3 IS 5 (La somma di 2+ 3 è 5)

Ogni programma termina con una dichiarazione di "END". Ogni istruzione è preceduta da un "label" (etichetta): in questo esempio 10 o 20.

La label è richiesta per classificare le istruzioni in ordine. Se vogliamo aggiungere una nuova istruzione, possiamo battere per esempio:

9 "PRINT" "THIS IS AN ADDITION" (Stampa "questa è un'addizione")

La nuova istruzione sarà inserita automaticamente nel programma prima di quella etichettata "10", ed il nuovo programma è:

```
9 PRINT "THIS IS AN ADDITION"      (Stampa "questa è un'addizione)
10 PRINT "THE SUM OF 2 + 3 IS" 2 + 3 (Stampa "la somma di 2 + 3 è" 2 + 3)
20 END                             (Fine)
```

Usando le labels (etichette), diventa possibile aggiungere istruzioni in qualsiasi momento.

Per flessibilità, è tradizionale etichettare inizialmente le istruzioni in successione come 10, 20, 30 ecc. così che possano essere aggiunte facilmente molte istruzioni.

L'istruzione 10 ha effettuato un'*operazione aritmetica.* Il BASIC può eseguire cinque forme di operazioni.

+ – * / ↑ **

+	e	–	sono gli usuali più o meno
*	e	/	rappresentano la moltiplicazione e la divisione
↑	o	**	rappresenta una potenza, a secondo di come la stampante del sistema stampa caratteri speciali.

In aggiunta, si possono usare parentesi per specificare raggruppamenti e l'ordine in cui le operazioni devono essere eseguite.

Questo semplice programma mostra pure che esistono due tipi di istruzioni:

- istruzioni eseguibili, come PRINT
- comandi del sistema, come END o RUN

Complichiamo un po' il programma:

```
10 LET A = 2                          (Poni A = 2)
20 LET B = 3                          (Poni B = 3)
30 LET S = A + B                      (Poni S = A + B)
40 PRINT "THE SUM OF A + B IS" S (Stampa "la somma di A+B è" S)
50 END                                (Fine)
```

Il risultato è lo stesso di prima. "A", "B" e "S" si chiamano *variabili.* L'istruzione "PONI" assegna un valore 2 ad A, e 3 a B, poi calcola A + B ed assegna il valore della somma ad S.

Le variabili nel BASIC devono comprendere una sola lettera, e possono essere seguite da un digit facoltativo.

In formule si ha: [variabile] = [lettera] ([digit]) dove () indica 0 o 1. (Questa si chiama "notazione BNF" o "Backus normal form", ed è un modo formale di descrivere la sintassi di un linguaggio).

La parola "LET" è un residuo del passato. Dei BASIC "puri" più vecchi la usano. È totalmente non necessaria, ed inutilmente d'impaccio. La maggior parte dei BASIC recenti esentano dall'uso esplicito di "LET". Le altre regole restano le stesse.

Come sono rappresentati i numeri? Il BASIC puro ammette sia i numeri interi che i decimali. Sfortunatamente il BASIC di molti microcomputers fornisce solo numeri interi. Questo è un inconveniente notevole.

In aggiunta, è essenziale verificare quanti digit vengono forniti. 6 digit sono un minimo assoluto, e non permetteranno la rappresentazione di quantità superiori a $ 9999,99.

LETTURA DEI VALORI

Nel BASIC ci sono due espressioni per leggere i valori: "READ" (leggi) ed "INPUT".

"READ" viene usato assieme all'istruzione DATA (dati) ed assegnerà ad una variabile un valore limitato come DATA.

Esempio:

```
10 DATA 2 (Dati 2)
20 READ A (Leggi A)
```

risulta A = 2

Possono essere elencate variabili multiple:

```
10 DATA 2, 3, 4, 5, 6     (dati 2, 3, 4, 5, 6)
20 READ A, B, C, D, E (leggi A, B, C, D, E)
```

"INPUT" legge un numero dalla tastiera. Il programma si ferma ed appare un "?" sul display. L'utente deve battere sulla tastiera un numero e determinare con un "carriage return" (carattere speciale). Allora il valore viene letto automaticamente dal programma, e poi esso procede.

Esempio:

10 INPUT A	(Ingresso A)
The computer types "?"	(Il computer scrive "?")
The user types 24 (CR)	(L'utente batte 24 (CR)
The program proceeds, with A now egual to 24	Il programma ora procede con A = 24

COMMENTI

Ora vorremmo rendere più chiaro il nostro programma scrivendo in esso dei commenti. Questo è possibile per mezzo dell'istruzione di REM:

```
10 REM THIS PROGRAM WILL DOUBLE A NUMBER
   (Ricorda questo programma raddoppierà un numero)

20 INPUT A                 (Ingresso A)
30 PRINT "DOUBLE IS" 2*A (Stampa "il doppio è" 2 × A)
40 GO TO 20                (Vai a 20)
```

La quarta istruzione è nuova. Essa è un'istruzione GO TO (vai a). Essa stabilisce che, invece di eseguire la prossima istruzione sequenziale, il programma ritornerà all'istruzione 20" ed eseguirà INPUT A di nuovo.

DOMANDA: *Quando si fermerà questo programma?*

RISPOSTA: *Da solo, mai. Dovrete interromperlo voi. Questo si chiama un loop (ci-*

clo) infinito. Normalmente in un programma non ci dovrebbe essere alcun loop infinito.

DOMANDA: *Cosa fa il seguente programma?*

10 REM SALES TAX COMPUTATION (Ricorda il calcolo della tassa sulle vendite).
20 INPUT P (Ingresso P)
30 REM USER TYPES NOW THE PURCHASE PRICE (Ricorda ora l'utente batte il prezzo di vendita).
40 T = P* 6,5/100
50 PRINT 'TAX AT 6,5% IS'T (Stampa "tassa al 6,5% è" T)
60 OUTPUT "PLEASE PAY" (P + T) (Uscita "prego pagare") (P+ T)
70 OUTPUT (Uscita)
80 GO TO 20 (Vai a 20)

DOMANDA: *Qual'è l'effetto dell'istruzione 70?*

RISPOSTA: *Questa istruzione inserisce una linea in bianco.*

NOTAZIONE SCIENTIFICA

Per non sprecare spazio (e tempo) sul foglio di carta, il BASIC stampa i numeri decimali in una forma normalizzata, chiamata notazione scientifica:

E + 05 rappresenta 10^5

(E viene chiamato l'esponente)

L'ESAME DELLE CONDIZIONI

Il vero potere di un programma non è tanto nella sua capacità aritmetica quanto nella sua capacità di prendere decisioni sui risultati dei test. Per avere questo risultato occorrono istruzioni speciali:

10 IF A = 1 THEN 50 (Se A = 1 quindi 50)

significa: se A è uguale a 1, allora esegui l'istruzione 50 ("GO TO 50").

L'istruzione IF. . . THEN. . .è abbastanza potente.

Possono essere usati sei operatori di relazione;

=	(uguale)
<	(minore di)
>	(maggiore di)
>=	(maggiore o uguale)
<=	(minore o uguale)
<>	(non uguale a, diverso da)

In aggiunta, l'"IF" può essere seguito da una completa espressione aritmetica e il "THEN" può essere seguito da un comando.

AUTOMATIZZARE I LOOPS

È frequentemente necessario ripetere un'operazione fino a che viene soddisfatta una certa condizione. Questo può essere fatto usando l'"IF"... "THEN".

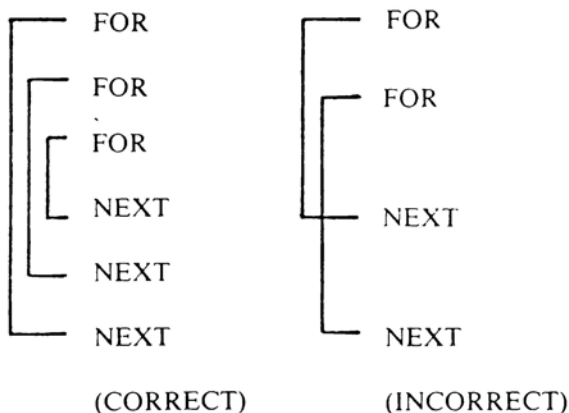
Comunque, se una variabile viene incrementata regolarmente, esiste un mezzo più efficace: questa è il "DO LOOP" realizzato con "FOR"... "NEXT".

Esempio:

```
10 REM PRINT NUMBERS FROM 1 TO 50 (Ricorda di stampare i numeri da
   1 a 50)
20 FOR A = 1 TO 50 (Per A = 1 a 50)
30 Print A (Stampa A)
40 NEXT A (Prossimo A)
50 END (Fine)
```

Questo programma assegna il valore "1" ad A, poi lo stampa. L'istruzione 40 ("NEXT A") automaticamente;

- incrementa di 1 A: A diventa uguale a 2
- forza un "branch to the FOR" (diramazione al FOR) cioè all'istruzione 20 in questo caso. Il programma esegue un loop fino a che A raggiunge 50. Quando A = 50 l'istruzione 40 fallisce poichè A eccede la propria gamma di valori specificata ("50"), e viene eseguita la prossima istruzione sequenziale: "END". Il programma termina.

Tali loops possono essere inseriti uno dentro l'altro purchè essi non si sovrappongano:

FUNZIONI INCORPORATE

È desiderabile fornire mezzi aritmetici in più, ma tutti i caratteri disponibili sono già utilizzati. Questo problema viene risolto fornendo funzioni a 3 caratteri:

INT ( ), SQR ( ), ABS ( ), SIGN ( )
INT è il valore intero: INT (−2,5) −2
SQR è la radice quadrata
ABS è il valore assoluto: ABS (—2.5) = 2.5
SGN è il segno: + 1 per un numero positivo
−1 per un numero negativo
0 per 0

LE SUBROUTINES

Una subroutine è un segmento di programma scritto come un'unità separata, che può essere usata ripetutamente nel programma principale, chiamandola. Un esempio:

(PROGRAMMA PRINCIPALE)

```
50    GO SUB 150 (Vai alla subroutine 150)
60    •••
150   (SUBROUTINE)
200   RETURN (Ritorno)
```

Il gradino 50 provoca l'esecuzione di un "jump" (salto) alla istruzione 150. La subroutine viene eseguita fino a che viene incontrato "RETURN". Quando si trova RETURN (Ritorno), l'esecuzione "torna al programma principale". La prossima istruzione da essere eseguita sarà l'istruzione 60, cioè l'istruzione che segue la chiamata della subroutine.

Concettualmente, la subroutine viene inserita durante l'esecuzione al posto della chiamata.

Questo evita il dover scrivere questa subroutine esplicitamente ogni volta che se ne ha bisogno nel programma principale. Una semplice "GO SUB" ottiene lo stesso risultato.

FUNZIONI DEFINITE DALL'UTENTE

Il BASIC permette ad un'utente di definire le proprie funzioni, usando un'istruzione DEF FCN (Definire Funzione).

DEF FCN (X) = [espressione]

LISTE ED ARRAYS

Il BASIC fornisce solo due strutture: liste uni-dimensionali e, arrays bidimensionali.

Il programma seguente leggerà 10 numeri battuti sulla tastiera e poi li stamperà in sequenza:

```
10 FOR I = 1 TO 10                    (Per I = 1 a 10)
20 INPUT L [I]                        (Ingresso L [I]
30 NEXT I                             (Prossimo I)
40 PRINT L [1]; L[2];...; L[10]       (Stampa L [1]...)
```

L'istruzione "FOR...TO" fissa un LOOP: "I" parte col valore 1, e viene incrementato dal gradino 30. Dopo il gradino 30, l'istruzione successiva è il gradino 10 fino a che I raggiunge 10. Quando ciò avviene, l'istruzione successiva alla 30 è la 40 (fine del loop).

ESEMPIO

Il programma seguente legge 10 numeri dalla tastiera, e poi calcola la somma:

```
20 S = 0
30 FOR I = 1 TO 10              (Per I = 1 a 10)
40 INPUT A [I]                  (Ingresso A [I])
50 S = S + A [1]
60 NEXT I                       (Prossimo I)
70 PRINT "THE SUM IS"; S        (Stampa "la somma è": S)
```

Il lettore dovrebbe verificare questo programma. S è la somma. Incomincia con un valore zero (gradino 20), poi viene incrementato del valore di ogni elemento successivo A [1] letto dalla tastiera (gradino 50) durante il loop (gradini 30-40-50-60), poi infine stampato (gradino 70).

ESEMPIO MODIFICATO

Il programma seguente prima stampa i 10 numeri, poi il totale T:

```
10 REM COMPUTE TOTAL OF SALES TODAY FOR 10 ITEMS
   (Ricorda calcola il totale delle vendite oggi per 10 articoli).
20 FOR I = 1 TO 10       (Per I = 1 a 10)
40 INPUT A [I]           (Ingresso A [I])
50 NEXT I                (Prossimo I)
60 PRINT "INDIVIDUAL SALES TODAY ARE: "(Stampa "Le vendite
   individuali di oggi sono)
70 FOR J = 1 TO 10       (Per J = 1 a 10)
80 PRINT [J];            (Stampa [J]);
```

```
90   NEXT J                      (Prossimo J)
100  REM COMPUTE TOTAL           (Ricorda calcolare il totale
     NOW                         ora).
110  T = 0
120  FOR K = 1 TO 10             (Per K = 1 a 10)
130  T = T + S [K]
140  NEXT K                      (Prossimo K)
150  PRINT "TOTAL IS"; T         (Stampa "il totale è"; T)
END                              (Fine)
```

UNA TABULAZIONE COMMERCIALE

Elenchiamo il numero di articoli venduti per tutte le dieci voci dell'inventario. L'espressione DATA elenca gli articoli venduti durante la giornata.

N [I] rappresenta il numero di articoli venduti per ogni voce I.

```
10   REM SET ALL N'S RO ZERO(Ricorda poni tutti gli N a
                             zero)
20   FOR I = 1 TO 10         (Per I = 1 a 10)
30   LET N [I] = 0           (Poni N [I] = 0)
40   NEXT I                  (Prossimo I)
50   READ Q                  (Leggi Q)
60   REM TEST FOR LIST ITEMS(Ricorda prova per elenco -
     SOLD EXHAUSTED          Voci vendute esaurite)
70   IF Q = 0 THEN 100       (Se Q = 0 quindi 100)
80   LET N[Q] = N [Q] + 1    (Poni N [Q] = N [Q] + 1)
90   GO TO 50                (Vai a 50)
100  PRINT "TOTAL SALES PER  (Stampa "i totali vendite per
     ITEM ARE"               per voci sono")
110  FOR I = 1 TO 10         (Per I = 1 a 10)
120  PRINT I, N [I]          (Stampa I, N [I])
130  NEXT I                  (Prossimo I)
200  DATA 2, 1, 1, 1, 8, 7, 3, 4, 5, 9, 1,(Dati 2, 1 ...)
     2, 1, 0
999  END                     (Fine)
RUN                          (Opera)
```

LE VENDITE TOTALI PER VOCE SONO

1	5	6	0
2	2	7	1
3	1	8	1
4	1	9	1
5	1	10	0

PRESTAZIONI PARTICOLARI

Il BASIC fornisce anche istruzioni per operare su matrici, stringhe, e files, come pure una specificazione del formato ("TAB") dell'output.

La maggior parte dei BASIC permettono anche espressioni multiple su di una linea singola, separate da &, ; , o/, come pure degnazioni multiple come x, y, z = 0.
Infine, REM è spesso sostituito da (*) o da commenti tra virgolette.

QUANT'È COMPLETO IL VOSTRO BASIC?

Sebbene il BASIC sia, in teoria, standard, non ci sono due BASIC completamente simili. Le maggiori differenze che si possono trovare, cioé le facilities che mancano più spesso sono:
- disponibilità di numeri decimali

Potrebbe essere sorprendente, ma la maggior parte dei "mini BASIC" forniscono solo aritmetica intera.
- numero di digit usati per rappresentare i numeri. 6 è un minimo assoluto. 9 può essere un numero. Ne sarebbero desiderabili di più.
- disponibilità di un file system, così che i dati ed i programmi possano essere recuperati automaticamente prelevandoli da un disk o altro supporto.
- sia file sequenziali che random (a caso) (file sequenziali sono usati sui nastri, i random files sui disk).
- istruzioni di formato (sull'output)
- piene capacità di manipolazione di stringhe
- in aggiunta, l'editor incorporato dovrebbe essere abbastanza potente da edit il programma convenientemente: inserire - aggiungere - modificare - ricerca di un modello, ecc.

Facilities addizionali che possono migliorare il BASIC sono:
- disponibilità di hardware, in floating point package (package a virgola mobile FPP). Questa scheda speciale accelera le operazioni aritmetiche di un fattore di 10 o più.
- CHIAMATA (CALL) dei programmi in linguaggio macchina: in questo modo i segmenti di programmà usati più frequentemente possono essere codificati in assembler, convertiti in codice object ("linguaggio macchina"), e usati dal programma in BASIC col risultato di un'altra velocità d'esecuzione.

In sommario, le deficienze trovate più spesso nei vari usi del BASIC sono capacità insufficienti di input-output, manipolazione di stringhe, e aritmetica decimale di alta precisione.

BASIC COMMERCIALE

"Basic Commerciale" (Business Basic) si riferisce ad un BASIC completo, equipaggiato con capacità addizionali di elaborazione dati (per la maggior parte text processing).

Capacità addizionali tipiche possono includere:
1-Numeri interi binari (spesso indicati dal simbolo %)
2- Aritmetica di precisione ampliata (precisione da 12 a 18 digit).
È importante ricordare che un computer tronca i numeri internamente, così che si possono perdere digits nelle operazioni aritmetiche. Il calcolo dell'errore di propagazione esula dallo scopo di questo libro. Comunque, 3 × 1/3 può non dare "1" come risultato, ma 0,999999999 per esempio: la maggior parte dei calcoli commerciali eseguono solo della semplice aritmetica. In tali casi, l'errore di troncamento è minimo, e spesso vicino a zero, o zero. Per una precisione accettabile nel risultato, è importante usare internamente
il più grande numero possibile di digit.

10 digits è probabilmente un minimo assoluto per applicazioni commerciali. 18 sarebbero altamente desiderabili.

ATTENZIONE: Prima di usare un "BASIC" per scopi commerciali, si raccomanda vivamente di eseguire un po' di "aritmetica del caso peggiore", e osservare la precisione dei risultati.
3-più formati
Nelle applicazioni commerciali, è essenziale generare reports o dati nel formato adeguato per la stampa. E' quindi importante la disponibilità di comandi adatti. Il BASIC standard come il FORTRAN, è celebre per la difficoltà che si incontra nel manipolare un testo, e nel formatting.
4-stringhe di caratteri bianchi
Una, facility importante nel posizionare un testo in una pagina sta nell'abilità di fissare stringhe di caratteri bianchi convenientemente.
5-ricerca di sub-string
Quando in una lista si va in cerca di un nome o di un codice, il programma sta frugando una stringa in cerca di una "substring". Comandi che automatizzino questa procedura agevolano molto, o persino rendono fattibili tali applicazioni. Qualsiasi aggiornamento in una lista, per esempio, andrà in cerca delle substrings così da metterle in ordine alfabetico, o da controllarne per la duplicazione.
6-assegnazione delle stringhe
Quando si manipolano stringhe, ci deve essere la capacità di maneggiarle come un dato qualsiasi, cioé dar loro un nome, assegnarle ad un'altra variabile e modificare il loro contenuto. La modificazione del contenuto può comportare troncamento, concatenazione (aggiungere un testo nuovo), inserzione, o sostituzione.

QUANT'È VELOCE IL VOSTRO BASIC?

L'efficienza del programma interprete che implementa il BASIC è il fattore chiave. Naturalmente, la velocità del microprocessore in sé gioca un ruolo determinante. Comunque, un interprete mai progettato sarà dieci volte più lento di uno professionale.

Molti costruttori si sono inseriti nel mercato con un "mini BASIC" trasandato che lavora correttamente ma lentamente, e che è incompleto. Essi stanno prendendo tempo per sviluppare una nuova versione di BASIC completa, o quasi, (o con miglioramenti). Quindi è difficile valutare la velocità di un BASIC.

Il modo in cui ciò viene fatto consiste nell'usare "*benchmark programs*". Un programma di riferimento è un programma tipico scritto dall'utente per le applicazioni che egli desidera. Facendo girare questo programma su vari computer differenti si ottengono cifre rigorose su cui basare il confronto. Questo è un modo efficiente per realizzare un confronto. Comunque può capitare che le istruzioni debbano essere "leggermente" modificate nel far girare lo stesso programma con vari interpreti di BASIC.

BASIC COMPILATO

La relativa inefficienza (da un punto di vista della velocità) di un interprete BASIC è intrinseca a tutti gli interpreti. Un *interprete* traduce ogni istruzione in formato eseguibile dalla macchina, e la esegue immediatamente. Una volta che una frase è eseguita, non rimane nulla, al di là del risultato dell'operazione.

Ora la maggior parte dei programmi usano dei *loops*, e li eseguono un grande numero di volte. In un loop, ogni frase viene tradotta, quindi eseguita n volte se il loop viene eseguito n volte. Questo è del tutto inefficiente dal punto di vista della velocità.

Un *compilatore*, d'altra parte, traduce una volta l'intero programma dell'utente nel formato eseguibile dalla macchina ("codice object"), che può poi essere eseguito in qualsiasi momento. Il codice object può avere come risultato un'esecuzione ripetuta di un loop, senza il bisogno di tradurre frasi ripetutamente: l'esecuzione è molto più veloce.

Però, una volta che è stata eseguita la traduzione, diventa difficoltoso aggiungere, modificare o correggere frasi e l'intero programma deve essere tradotto di nuovo, un processo troppo lungo. Per questa ragione, i compilatori non sono normalmente usati in un'ambiente *interattivo*. Infatti, il BASIC fu sviluppato specificatamente come una versione *interpretativa* del linguaggio FORTRAN (il FORTRAN è compilato). Ci sono ora sistemi che forniscono un BASIC compilato in aggiunta al regolare interprete. I benefici in velocità possono essere molto consistenti (parecchie volte più veloce, a seconda del programma e dell'efficienza del compilatore).

Questa facility è, valida per programmi finiti, che non devono essere cambiati per un po'. L'interprete è ancora altamente desiderabile per qualsiasi tipo di sviluppo.

LE LIMITAZIONI DEL BASIC

Il BASIC è di gran lunga il linguaggio più facile da usare, ma non è completo.

1 - Le sue capacità aritmetiche sono limitate dal numero limitato di funzioni in-

corporate, precisione limitata, e lentezza di esecuzione. In aggiunta, può manipolare solo arrays-bi-dimensionali.

2 - Le sue capacità di manipolazione delle stringhe sono minime, rendendo difficile la manipolazione di liste di informazioni come catene di caratteri, o strutture più complesse.

3 - La sintassi del linguaggio non permette di costruire strutture a blocchi complesse ed il risultato è una limitazione consistente della complessità dei programmi che possono essere scritti.

ALTERNATIVE AL BASIC

Storicamente, l'unica "alternativa" al BASIC è stato il PL/M. Il PL/M è brevettato dalla INTEL Corporation. PL/M sta per *"Programming language for Microprocessors"* (Linguaggio di Programmazione per Microprocessori). È un linguaggio compilato derivato dal linguaggio PL/I della IBM o più precisamente, dal dialetto XPL.

Come linguaggio ad alto livello il PL/M offre i vantaggi classici rispetto ai linguaggi a livello macchina: i programmi sono più semplici e più veloci ad alto livello: l'esecuzione è consistentemente più lenta, ed i programmi sono più lunghi di quelli in linguaggio a livello macchina.

Il PL/M ha il merito di essere il primo linguaggio ad alto livello preparato per microprocessori. La maggior parte dei costruttori forniscono una versione di PL/M, con vari nomi quali MPL, XPLM o SMPL.

Il PL/M viene veramente usato per favorire un certo numero di applicazioni industriali. Comunque il PL/M è relativamente complesso, difficile da usare, ha solo aritmetica intera, ed è un compilatore.

Il PL/M di solito non viene usato per applicazioni personali o commerciali.

La prossima domanda è: "c'è un'alternativa migliore al BASIC?". C'è: l'APL.

APL

L'APL è un linguaggio originariamente inventato da Iverson, e che ha guadagnato un riconoscimento lento, ma sempre crescente. *L'APL è probabilmente uno dei linguaggi più adatti alle applicazioni personali e commericali.* Le ragioni:

1 - L'APL è per autodidatta

Le regole della sintassi dell'APL sono talmente semplici e chiare che chiunque può incominciare a scrivere un programma in un'ora o due, stando semplicemente seduto ad un terminale e "parlando alla macchina". L'APL è altamente interattiva, e gli errori di sintassi vengono scoperti immediatamente nel momento stesso in cui vengono battuti sulla tastiera.

2 - L'APL è molto potente

L'APL usa operatori speciali e può eseguire operazioni altamente complesse su qualsiasi struttura pratica. Può persino trasporre una matrice per mezzo di un operatore speciale (una matrice n-dimensionale!). Programmi altamente complessi

possono essere espressi in appena poche istruzioni. In particolare, si è trovato che l'APL è quasi il linguaggio ideale per applicazioni commerciali, ed è spesso usato in scuole commerciali.

3 - L'APL è un vero "linguaggio lambda"

Senza esprimere in dettaglio i principi di calcolo lambda, l'APL permette all'utente di scrivere programmi di illimitata complessità, comunicando attraverso strutture o variabili predefinite.

Naturalmente l'APL ha anche svantaggi:

1 - Usa un insieme di caratteri speciali

Gli operatori APL usano una varietà di simboli come □, e 2, non disponibili su molti display o stampanti (cominciano a diventare disponibili ora).

2 - I programmi APL sono così condensati da essere talvolta indecifrabili

Questo è il risultato del grosso potere del linguaggio. Questo problema può essere semplicemente risolto per mezzo di abitudini di programmazioni migliori e buona documentazione.

3 - È molto difficile sviluppare un interprete APL efficiente

Questo problema è stato risolto. Vedi il nostro riferimento Z10.

Esempi semplici

- Il seguente sommerà due "vettori"
  1 2 3 4 5 + 2 3 4 5 6

risulta:
  3 5 7 9 1 1

- aritmetica semplice
  2 × 3

stampa: 6

- 1 + 2 + 3 + 4

stampa: 10

- 'HELLO'

stampa: HELLO

- operatori potenti
  - !6
    rappresenta il 6 fattoriale, cioé 1 x 2 x 3 x 4 x 5 x 6
  - ⌊ 2.3
    è la (base) di 2,3 cioé 2
  - ⍳ 5 (iota 5)
    genera il vettore 12345

Un esempio complesso

Il fattoriale di n è definito da:

n! = 1 x 2 x 3 x ... x n

È semplicemente il prodotto dei primi n numeri. Il seguente programma APL lo calcola:

```
∇ Z ← FAC N
[1] → 4* 2N = 0
[2] Z ← N* FAC N −1
[3] → 0
[4] Z ← 1
∇
```

La prima linea di questo programma è una definizione di "funzione". La funzione è avviata e terminata dal simbolo speciale ∇ (delta).

Il nome della funzione è FAC.

Il suo parametro è la variabile N.

Il suo risultato è la variabile Z.

L'istruzione [1] stabilisce: GO TO [4] if N = 0 (la combinazione * 2 può essere letta come "if" = se)

[2] stabilisce: prende il valore di N moltiplicato per FAC di (N-1).

FAC (N-1) è una funzione chiamata all'interno della funzione FAC. Ciò si chiama una chiamata *recursiva*, dove una funzione chiama se stessa. Questa determina che la funzione FAC viene re-introdotta per l'esecuzione, con un parametro (N-1).

[3] significa "GO TO step 0" (vai alla fase 0), cioé esci (ritorna dalla funzione). Questo pone termine all'esecuzione della funzione. A questo punto Z dovrebbe contenere il valore del fattoriale.

[4] assegna il valore 1 a Z.

[5] è la fine della funzione. Esso pone anche termine all'esecuzione della funzione.

ESERCIZIO: FAC 3 dovrebbe avere come risultato Z= 6. Provalo a mano e segna l'esecuzione passo dopo passo.

COSA PENSARE DEL "LINGUAGGIO NATURALE"?

Naturalmente, il linguaggio più desiderabile per comunicare con un computer è il linguaggio parlato ordinario, l'inglese o l'italiano per esempio.

Sarebbe il modo più pratico per l'utente "programmare" direttamente in inglese.

Durante i bei giorni della NASA, e della gara spaziale, il governo degli Stati Uniti fondò dei programmi di ricerca vasti per determinare la possibilità di usare l'inglese, o un adatto sotto-insieme, come linguaggio di programmazione. I risultati furono totalmente inequivocabili: l'inglese ordinario non poteva essere usato.

Apparentemente, un certo numero di utenti non sono ancora a conoscenza di questo fatto, ed alcune società propagandano regolarmente la "programmazione direttamente dall'inglese!".

Il problema è l'ambiguità. Il linguaggio parlato non è abbastanza preciso da essere inequivocabile, e fa affidamento sul contesto per specificare il significato della frase. Il contesto può essere sensoriale (gesto, un odore, ecc.) o sintattico, dipendente da ciò che è stato detto prima, o che sarà detto dopo. Il processing necessario per risolvere tali ambiguità semplicemente esaurirebbe tutte le risorse di un computer, senza alcuna garanzia di successo. Perciò un tale approccio del problema è, nel migliore dei casi, totalmente inefficiente.

Deve essere usato un sotto-insieme ben definito della lingua inglese con una "sintassi pulita". Questo si chiama linguaggio di programmazione.

È stata sviluppata una varietà di linguaggi di programmazione che vanno a genio a classi specifiche di utenti, a seconda della loro preparazione. Si è trovato che il BASIC e l'APL sono linguaggi facili da usare. Questo spiega il successo del BASIC nel mondo dei microcomputer.

Riassumiamo: *non c'è speranza di vedere un computer capire l'inglese ordinario nel prevedibile futuro: qualsiasi pretesa di* "Programmare direttamente in inglese" è deliberatamente ingannevole. Tali sistemi usano semplicemente un qualche linguaggio di programmazione, con le limitazioni sintattiche usuali.

Questo non esclude la possibilità di "parlare al computer" usando un sotto insieme ben definito della lingua inglese, e regole fisse. In questo caso, si sta semplicemente definendo un altro linguaggio di programmazione ad alto livello.

SOMMARIO DELLE ISTRUZIONI DEL BASIC

CHANGE A TO A	(Cambia A in A). Immagazzina il codice ASCII dei caratteri in array A, insieme alla lunghezza della stringa.
DATA	Usato con READ per immagazzinare valori di variabili, separati da virgole.
GO SUB N	(Vai alla subroutine N). È una chiamata di una subroutine. La subroutine è eseguita fino a RETURN.
END	Ultima espressione in un programma. Può essere facoltativo.
FOR I = A TO B STEP C	Istruzione di loop. La parola step è facoltativa.
DEF FNX (Y)	Definizione di funzione. X è il nome della funzione Y è l'argomento.
DIM A (X)	Dimensione per l'array A (probabilmente bidimensionale)
GO TO n	Specifica l'istruzione n come la prossima da essere eseguita.
IF x THEN y	Viene effettuato il test logico x. Se ha successo, viene eseguita l'istruzione y. Altrimenti viene eseguita la prossima istruzione sequenziale.
INPUT	Fa si che vengano letti i dati dalla tastiera fino ad un (CR) carriage return.

LET	(spesso facoltativo) Espressione di assegnazione di un valore. Assegnazioni multiple di solito possono essere stringate su una linea.
MAT	Istruzioni di matrice.
PRINT	Stampa valori o un testo, separati da virgole o punto e virgola.
NEXT I	Fine di un loop. Incrementa I e ritorno alla istruzione FOR precedente.
READ	Usato con DATA per assegnare un valore alle variabili.
REM	Commento inserito in programma (non eseguibile).
RESTORE	Restituisce i DATA statements alla loro prima voce.
RETURN	Ultima istruzione eseguita in un subroutine. Causa un "ritorno" alla istruzione che segue la GO SUB.
STOP	Ferma l'esecuzione del programma. Sostituisce REM in alcuni sistemi.

ALTRE FACILITIES DEL BASIC

MAT:	Una varietà di istruzioni a matrice.
FUNCTIONS:	Funzioni predefinite quali ABS(X), RND(X), SGN(X), SQR(X), SIN(X), COS(X), TAN(X), LOG(X), EXP(X), TAB (X).
FILE S facilities:	Per dare un nome ai files leggere, scrivere, inserire, aggiungere, ripristinare i file pointers. I files possono ssere ASCII, binario, sequenziale, in formato o ad accesso casuale.

CAPITOLO 7

BUSINESS COMPUTING

INTRODUZIONE

Per la prima volta il progresso della tecnologia fa si che sia possibile godere i benefici di un computer in un piccolo ambiente commerciale per meno di cinquemila dollari. *Ma è realmente vero?* La risposta è "si ma...". Lo scopo di questo capitolo è giustificare il "si" e descrivere il "ma".

Può, un sistema a microcalcolatore a basso costo fornire vere capacità di calcolo commerciale? Si.

C'è qualche sistema disponibile attualmente che lo fa? No.

Si vedrà che la deficienza essenziale dei sistemi microcalcolatori attuali non è a livello *hardware* ma a livello *software*. Questa è sempre stata la situazione, persino da quando i computers furono introdotti, e la storia si è ripetuta ogni volta che una nuova generazione di hardware è stata introdotta. Si vedrà che l'hardware necessario ad eleborare efficientemente un certo numero di applicazioni commerciali può veramente essere acquistato per cifre da 5.000 a 20.000 dollari. Comunque, il software sta proprio ora cominciando a diventare disponibile. Naturalmente esistono molte offerte in funzione delle capacità che uno vuole acquistare, e queste saranno studiate.

Perciò, in primo luogo passeremo in rassegna le applicazioni classiche dei computers nel commercio, così da definire le capacità di processo, richieste per raggiungere specifici obiettivi commerciali. Affinché un businessman possa fare una scelta conveniente di un sistema microcalcolatore, *è imperativo che egli comprenda* le alternative, tra le varie soluzioni disponibili attualmente poiché *non esiste il "migliore"*. La scelta è molto simile alla scelta di una nuova automobile o di una nuova macchina complessa in funzione di una specifica applicazione che si intende ottenere. Non c'è scelta per scopi generali che si adatti a tutte le applicazioni.

Capire l'*hardware* necessario e l'hardware disponibile è una cosa relativamente semplice. Il problema più complesso e difficile è capire le capacità *software* richieste. Qui è dove la maggior parte delle persone sbaglia nell'acquisto di un sistema commerciale. Questi errori sono di solito più costosi di quelli fatti sull'hardware. Tipicamente gli investimenti nel *software* di un sistema diventano rapidamente gli

investimenti dominanti. Un sistema inadeguato limiterà la possibile crescita delle capacità del sistema, e forse anche la crescita degli affari. Il passaggio ad un sistema differente potrebbe essere costoso e distruttivo. Per queste ragioni, il lettore è vivamente incoraggiato a studiare e comprendere i concetti software come quelli hardware che saranno presentati.

APPLICAZIONI DEI COMPUTERS NEL CAMPO COMMERCIALE

Ogni azienda ha bisogno innanzitutto di tenere un certo numero di *files*. I files più conosciuti sono: acconti ricevuti, acconti da pagare, inventario, registro generale. Files addizionali che possono essere di solito desiderabili sono: personale, lista dei clienti, lista degli indirizzi per l'invio di materiale pubblicitario, liste di ordinazioni arretrate, lista delle vendite, lista dei venditori, situazione di cassa, beni dell'azienda, ed altre.

Queste liste vengono manipolate sia a mano (di solito da un contabile) o con l'aiuto di dispositivi elettromeccanici, o dal computer, o da una combinazione di ciò di cui sopra.

In aggiunta per mantenere aggiornati i files ogni azienda applica tecniche *procedurali* specifiche per opere. Per esempio, *un programma* per libro paga opererà sul *file del personale* e genererà *resoconti del libro paga,* come pure verifiche sulla stampa. Un *programma di tassa* opererà sui *resoconti di vendita* o sul *file del personale* per produrre i resoconti di *tassa richiesti. Un programma di procedura di transizione* manipolerà aggiornamenti di files specifici, ed il cambiamento o introduzione di nuovi dati. Un esempio tipico è una nuova vendita: il programma di transizione utilizzerà il file inventario, il file dei fornitori, il file dei clienti, e forse altri. Si aggiornerà, e stamperà i rapporti.

In modo simile una *procedura di spedizione* manipolerà spedizioni entranti e li introdurrà nel file inventario, controllerà l'esistenza di ordinazioni arretrate ed aggiungerà le registrazioni nella lista degli acconti da pagare.

Ogni *pagamento ricevuto* aggiornerà la lista degli *acconti ricevibili* e la lista della *situazione di cassa.*

In fig. 7-1, in basso, sono illustrate queste liste ed i tipici programmi di processing. Le frecce indicano l'effetto di un'azione specifica sui vari files del sistema.

In aggiunta ai programmi principali che sono indicati in figura sotto forma di cerchi, deve essere disponibile un certo numero di programmi addizionali così da produrre resoconti utili. Queste facilities addizionali richieste saranno descritte più dettagliatamente nelle prossime pagine.

È importante notare che il *principio è abbastanza semplice*:

1 - I files devono essere creati e mantenuti aggiornati.

2 - Dovrebbero essere disponibili programmi per fare da interfaccia tra l'utente ed i files, e per fornire le funzioni operative richieste.

Fig. 7-1: File e programmi commerciali.

Sfortunatamente in un sistema commerciale reale, questa è solo *una parte del processing* necessario. Infatti, nella maggior parte delle aziende, è abbastanza semplice la conservazione diretta di un *singolo* file. La maggior parte del processing necessario è dovuto all'esecuzione simultanea di *riferimenti incrociati* tra i vari file ed *aggiornamento automatico* di molteplici files.

Osserviamo un esempio. Viene ricevuta un'ordinazione per posta. Viene elaborata dal programma *manager delle transazioni* sulla sinistra dello schema. La vendita viene introdotta nel *file delle vendite* della giornata. Ora si svolge una complessa serie di eventi. Come risultato di questa introduzione, il nome del cliente verrà aggiunto automaticamente al *file dei clienti.* In aggiunta il suo nome verrà probabilmente codificato in funzione dell'acquisto che ha fatto o dell'ammontare dell'acquisto o la sua posizione d'impiego. In aggiunta, il suo nome verrà controllato per informazione su crediti prima che l'ordinazione venga elaborata. Purché la vendita non sia "proibita" dal programma *"manager dei crediti"*, il prossimo passo sarà l'accettazione dell'ordinazione. Nel file dell'*inventario vendibile* ora viene verificata la disponibilità degli articoli ordinati. In questo esempio vengono ordinati tre articoli AB e C. A e B sono a stock. C non lo è. Come risultato, viene generata una *bolletta* per il cliente (vedi la scatola all'estrema sinistra dell'illustrazione) *ed un back order* (ordinazione non eseguita, arretrata). Il back order è aggiunto alla vista dei *back order.* Nel nostro esempio, l'articolo B è disponibile a stock. Comunque sono rimasti solo quattro articoli B a stock. *La lista dell'inventario* è strutturata con un campo speciale che specifica il livello della riordinazione. Il livello di riordinazione dell'articolo B è quattro. Come risultato seguente a questa transazione, un back order a riordinazione saranno generati per l'articolo B per una quantità standard di 25 articoli (il numero 25 era specificato, nel file d'inventario). L'indirizzo del venditore è ottenuto dal *file dei venditori* usando il numero del venditore quale indica per la lista (a sinistra dello schema).

Questa semplice operazione di vendite ha richiesto l'uso di cinque files all'interno del nostro sistema e di vari programmi di processing. Per business specifici, può essere persino necessario aggiornare, verificare, o modificare files addizionali, o eseguire funzioni di processing addizionali. Dovrebbe essere chiaro da questo esempio che, per essere veramente utile, un sistema commerciale deve fornire modi per accedere ad una varietà di files e di modificarli ed elaborarli comodamente. In aggiunta esso deve fornire un meccanismo per eseguire le funzioni richieste automaticamente e non manualmente.

Sfortunatamente si vedrà che la *maggioranza dei così detti sistemi commerciali che utilizzano i microcomputers, disponibili sul mercato al giorno d'oggi, non eseguono un tale servizio completo.* Essi di solito forniscono la gestione di un file singolo e non automatizzano completamente il processo completo di transazione. Molto deve essere fatto "a mano".

WORD PROCESSING

"Elaborazione di Parole" si riferisce all'operazione di una macchina da scrivere computerizzata, dove l'utente può facilmente cambiare, modificare o dare forma al testo. Esso richiede un programma "editor", che è una prestazione standard sui computer tradizionali. Il costo del processore è diventato così basso che può essere dedicato ad una funzione quale il processing di parole così che i processori di parole "a sé stanti" si stanno moltiplicando. La maggioranza usa una Selectric o una macchina da scrivere simile. Per contrasto, i sistemi commerciali offrono l'opzione di display o multi-terminal.

USO DI UN SISTEMA COMPUTERIZZATO

Ora, usiamo un microcomputer in-house per una semplice operazione. Specificheremo il tipo di programma, e le nostre scelte in risposta a scelte o domande che appaiono sullo schermo del terminale CRT.

Inizialmente, il sistema visualizza un "menu". Un "menu" è semplicemente una domanda con scelta multipla. La domanda formulata dal computer è evidenziata con uno o più "caratteri di sollecito" (qui, "..."), destinati ad indicare che il microcomputer sta aspettando una risposta.

Fig. 7-2: Un "menu".

È stato scelto il "programma commerciale". Il sistema lo dovrebbe caricare automaticamente prelevandolo dal disk. Appare ancora un elenco di opzioni.

Specifichiamo gli "acconti ricevibili". A questo punto, il sistema può richiedere che sia inserito un altro disk. Supponiamo di no, e andiamo avanti.

Specifichiamo una nuova vendita, ed il sistema richiederà tutti i dati necessari per registrare la transazione, genererà una bolletta e più tardi aggiornerà tutti i files

in connessione ad essa come quelli di banca, acconti ricevuti, inventario, lista dei clienti. Il dialogo diventa ora altamente interagente con il sistema che richiede tutti i dati necessari. La transazione è ora completata.

Fig. 7-3: Il "submenu" commerciale

Il modo in cui si interagisce col sistema dovrebbe essere chiaro ora. Il programma formula tutte le domande necessarie, mettendo in risalto una certa materia. In

Fig. 7-4: Il file degli acconti ricevibili.

aggiunta, vedremo che esso dovrebbe pure verificare la validità dei dati che vengono introdotti (nessun errore grossolano). Alla fine dovrebbero esser stampate automaticamente le bollette e più tardi dovrebbero essere aggiornati i files interessati.

Esaminiamo ora i requisiti effettivi in maggiore dettaglio.

Fig. 7-5: Registrazione di una nuova vendita

Fig. 7-6: Registrazione della vendita, continuazione

Fig. 7-7: Fine della registrazione di vendita

I REQUISITI DI UN SISTEMA COMMERCIALE

Analizzeremo qui i requisiti di un sistema commerciale in termini dei *files* essenziali che devono essere conservati e delle funzioni di *processing essenziali* che devono essere eseguite.

Acconti ricevibili

Questo è essenzialmente il file che contiene una copia di tutte le bollette *generate* dal sistema. Naturalmente il file non contiene la copia reale, ma la quantità minima necessaria di informazioni possibili da immagazzinare, che permette al sistema di generare realmente una bolletta completa. Di solito, esso immagazzina la data della transazione, il nome e l'indirizzo del cliente, il punto di spedizione, informazioni di vendita quali il venditore, come è stato spedito, quando è stato spedito, e dettagli specifici degli articoli venduti. Può non essere necessario immagazzinare entro questo file degli *acconti ricevibili* tutte le informazioni che appaiono su una bolletta usuale. Se esiste un *file delle vendite*, tutte queste informazioni vengono immagazzinate in esso, e sarà necessario accedervi frequentemente e dovrà essere *elaborato efficientemente.*

Qualsiasi computer per effettuare un'elaborazione efficiente delle informazioni richiede che tutti gli elementi dentro un file siano di *lunghezza uguale*. Per questa ragione, tutti i files che vengono elaborati spesso, o che lo sono da programmi complessi, usano annotazioni o "block" di *lunghezza fissa.* Spazi prefissati possono essere assegnati ad informazioni essenziali quali la data, il nome, la somma dovuta, codice della transazione o del cliente, numero della bolletta. La presenza del numero della bolletta permette all'utente del sistema di accedere al resto di questa informazione posta nel *file delle vendite* o *nel file delle bollette*. In gergo di computer, la presenza di un numero usato per accedere ad un'informazione che è immagazzinata altrove viene chiamata **pointer.** Il numero di bolletta è il pointer per la bolletta reale. In gergo commerciale, questo è part del *audit trail* (pista di audit).

Il *file* degli acconti ricevibili deve essere distinto dal *programma* degli acconti ricevibili. Il *file* degli acconti ricevibili è semplicemente la *lista* degli acconti. Per l'utente commerciale è facile valutare i vantaggi e gli svantaggi del formato di questo file. Un requisito tipico è che esso contenga in un modo facilmente accessibile, tutti i settori di informazione di cui l'utente commerciale necessita frequentemente.

Il *programma* degli acconti ricevibili è responsabile della manipolazione di questo file, lo aggiorna e genera il rapporto richiesto. Deve anche generare rapporti specializzati come la pubblicazione degli acconti più vecchi di 30, 45, 60 o 90 giorni (questo è chiamato "aging"). Questo programma può persino essere responsabile della generazione di *reminder notes* (note di sollecito). Comunque, il programma di notificazione dei solleciti può essere un programma separato. In tal caso il programma degli acconti ricevibili verrebbe usato per generare un file di *acconti in ritardo*. Questo file sarebbe poi usato a sua volta dal *programma di notifica dei solleci-*

ti così da generare solleciti personalizzati a tutti i clienti elencati nel file dei ritardi di pagamento. Sia il separare le funzioni in programmi singoli sia l'integrarle entro un programma unico ha una piccola incidenza sul valore di questo sistema. È in gran parte una questione di convenienza per il progettista del sistema. Il punto importante è *che siano disponibili tutte le facilities.*

Acconti pagabili

Il file degli acconti pagabili è essenzialmente una lista di tutti i debiti o bollette ricevute dall'azienda. Di solito, ogni volta che viene introdotto un ordine di "OK pagare", il programma che gestisce gli acconti pagabili stamperà automaticamente assegni di pagamento per la merce ricevuta. Di solito l'assegno viene stampato sia ad una data specifica, o anche ad una data programmata quale il trentesimo giorno dalla ricezione della bolletta. (Un buon programma di verifica della stampa dovrebbe anche verificare che il bilancio di cassa nel conto bancario sia sufficiente a coprire le spese!).

Inventario

Non c'è un file di inventario ottimale, poiché l'informazione d'inventario è differente a seconda di necessità commerciali specifiche. Per questa ragione, la maggior parte dei files d'inventario general purpose implicano un vasto numero di categorie. Non tutte le categorie saranno usate dall'azienda. La non disponibilità di una categoria può essere sentita come uno svantaggio da alcuni utenti. La disponibilità di troppe categorie, d'altra parte significa che viene sprecata una considerevole quantità di spazio dentro il sistema. Ciò si traduce in un numero relativamente più piccolo di voci che possono essere introdotte nell'inventario. Comunque col costo delle memorie in continua diminuzione, vengono offerte sempre più categorie, per la maggior parte dei tipi di affari, anche se alcune di esse non saranno mai usate. Si dovrebbe ricordare che la grandezza del file d'inventario è limitata dall'immagazzinamento fisico disponibile, quale la grandezza di un diskette.

Informazioni tipiche che possono essere incluse in un file d'inventario sono le seguenti:

CODICE - NO DELL'ARTICOLO - DESCRIZIONE DELL'ARTICOLO - POSIZIONE DI IMMAGAZZINAMENTO - NUMERO DISPONIBILE - NUMERO DEL VENDITORE - PREZZO DI FORNITURA - PREZZO DI VENDITA - DATA DELL'ULTIMA VENDITA - MINIMA QUANTITA' PER RI-ORDINAZION.

Di solito devono essere forniti come minimo da 64 a 128 byte per una tale entrata. Usando tale formato, possono essere immagazzinati in un normale diskette da 1800 a 3600 voci.

Il *programma* di controllo di inventario deve fornire molte funzioni. Deve fornire facilities di gestione generale dell'inventario:

- conservazione completa dell'inventario, includendo l'aggiornamento automatico di qualsiasi categoria d'informazione entro il file.
- annotazione degli ordini di vendita.
- annotazione degli ordini d'acquisto.
- storia delle vendite.
- back order automatizzati (ordini arretrati, non eseguiti).
- lista per quantità, classe, costo, venditore, articolo, numero d'articolo, giorno di vendita.
- ricerca delle quantità minime.
- aggiornamento selettivo.
- rapporti dell'attività.
- liste d'inventario in funzione delle combinazioni dei criteri.

Come indicazione sommaria, la gestione d'inventario minimo scritto in BASIC richiederà 10K per parole di memoria (qui per tutti gli scopi pratici, una "parola" è un "byte", nel caso di microprocessori a 8 bit). Un programma più generale ne richiederà senza dubbio 90 o più. Poiché la memoria centrale di un microprocessore non è mai più grande di 64K viene usata una tecnica di *overlay*, così che un programma BASIC anche grande possa essere fatto operare su una memoria principale più piccola. Un *overlay* consiste nell'eseguire una parte del programma e poi portare nella memoria un'altra parte del programma e dovrà imprimerla al segmento della parte precedente che era stato immesso nella memoria principale e che non è più necessario, e così via. Perciò il programma BASIC completo interamente nella memoria non è mai completamente residente nella memoria in un sol pezzo. Pezzi di esso vengono portati nella memoria centrale a seconda della necessità. Naturalmente questo riduce l'efficienza del processing. Comunque, se le overlay sono scritte intelligentemente, l'impatto sull'efficienza è ragionevolmente piccolo.

Aggiornamento

È importante notare ancora una volta, che, tecnicamente, l'aggiornamento di un file d'inventario può essere realizzato manualmente. L'utente può esaminare la lista degli articoli nell'inventario e modificare qualsiasi delle annotazioni, come ad esempio il prezzo unitario. Comunque, il valore reale di un sistema calcolatore sta proprio nell'automatizzare l'aggiornamento di informazioni identiche in molti file. Perciò, un sistema commerciale completo dovrebbe aggiornare automaticamente il file di inventario, ogni volta che viene cambiata un'informazione rilevante in qualche altro posto. Per esempio, se fosse cambiato il prezzo unitario del prodotto, il file d'inventario dovrebbe essere aggiornato automaticamente come pure qualsiasi altro file dove debba risiedere il prezzo.

Mailing List

La mailing list (lista degli indirizzi) è spesso trascurata in un sistema commerciale. Questo perchè la maggior parte dei sistemi commerciali forniscono una *registrazione completa* delle vendite (lista della vendite). In tale registrazione di vendite, sono annotati il nome del cliente come pure tutte le informazioni interessanti alla transazione. Perciò a prima vista, potrebbe sembrare che il sistema abbia tutte le informazioni necessarie per listare i clienti per un'azione promozionale. Questo è tecnicamente corretto. Tuttavia, nella pratica, *tali liste di vendita non sono utilizzabili come mailing list per un uso immediato.*

Per brevi mailing (lista di indirizzi corte) la lista delle vendite può in realtà essere elaborata da un programma specializzato che genererà i nomi e gli indirizzi e li stamperà. Per *liste lunghe* (come diverse migliaia di nomi) un'elaborazione diretta della lista delle vendite è completamente non pratica. Per prima cosa, sarebbe lenta, poiché l'informazione nella lista delle vendite è strutturata in blocchi e solo poche annotazioni devono essere recuperate da ciascun blocco. Secondo, le informazioni nella lista delle vendite sono normalmente voluminose così che su un singolo mezzo fisico quale un diskette risiederanno, relativamente pochi nomi. Perciò l'elaborazione della lista delle vendite, richiederà un tempo consistente ed un frequente cambio manuale di diskette. In breve, generalmente è inaccettabile.

Per una gestione efficiente della mail list, è pure imperativo che l'utente *codifichi* ogni transazione nel momento in cui viene eseguita. Per esempio, l'utente dovrebbe codificare il tipo di affare, o il tipo di individuo coinvolto nella transazione. Egli dovrebbe codificare la natura degli articoli acquistati, e la portata della vendita. In questo modo, può essere usato un codice compatto che può essere utilizzato come *criterio di selezione* entro la lista. Una mailing list efficiente è creata elaborando la *lista dei clienti* o persino la *lista delle vendite* periodicamente. Deve essere mantenuta ordinata sia alfabeticamente sia mediante un codice postale o mediante uno speciale codice dell'utente. In questo modo ogni volta che viene introdotto un nome nuovo nella lista, può essere verificato immediatamente ed efficientemente la possibile esistenza di un doppione. Anche così, una mailing list tipica, ridotta al nome, indirizzo, e codice conterrà un gran numero di caratteri per ogni annotazione. Diamo un'occhiata a ciò; conterrà il nome di battesimo, le iniziali dei nomi secondari, il cognome, titolo, nome della compagnia, della divisione o del fermo posta, via o cassetta dell'ufficio postale, città, stato, codice postale. In aggiunta conterrà un codice di 5-20 digit per un recupero efficiente dei nomi. Risultato: forse 100 caratteri per annotazione. Assumiamo che 80 caratteri siano sufficienti per i nostri scopi. Un tipico blocco su un diskette contiene 256 caratteri. Nel nostro esempio un tale blocco conterrebbe solo 3 nomi. Un tipico diskette fornisce circa 1200 di tali blocchi, e perciò immagazzinerà circa 3600 nomi.

Questo non è sufficiente per la maggior parte delle mailing lists. Un programma di mailing list efficiente dovrebbe essere capace di manipolare in ogni caso da 1.000 a

10.000 o 50.000 nomi. Chiaramente un tale sistema sarebbe molto ingombrante da usare, poiché i diskette dovrebbero essere cambiati manualmente di frequente fino a che non è stata elaborata la lista completa. Questo può essere non accettabile.

I microcomputer, come risultato di questa limitazione, ristretta ulteriormente dalla quantità limitata di memoria a dischi, impongono la necessità di un file di mailing list *strutturato in maniera speciale*. Viene creato un blocco indice, che contiene solo pochi byte per annotazione. Esso contiene il nome o codice del cliente, ed un pointer del cliente, di solito un numero. In questo modo può essere ristretto ad un massimo di 20-30 caratteri. Di solito verrà assegnato un intero diskette a questo indice della mailing list. Ogni volta che deve essere eseguita una selezione, deve essere montato solo il diskette contenente il *blocco indice*. Come esempio, la selezione della mailing list si fa per tutte le compagnie nel campo di distribuzione, che hanno ordinato parti del valore superiore a 100 dollari negli ultimi tre mesi. Il programma di selezione otterrà tutti i suoi dati dal *file indice* e genererà una *lista di numeri* di clienti che soddisfano i criteri specificati. Poi questa lista può essere usata da un semplice *programma di stampa* della mailing list che stamperà gli indirizzi presi dal file dei clienti o dal file delle vendite, a seconda di come vengono selezionati dalla lista dei numeri dei clienti. Una volta eseguita la selezione, il programma incomincierà a generare messaggi come: *"per favore carica il diskette numero 2-5"*. Questo diskette contiene il primo blocco di annotazioni che questo elaboratore di mailing list vuole stampare. Successivamente si dovranno montare ancora un certo numero di diskette nel sistema, se la mailing list è lunga, considerando le limitazioni inerenti a ogni diskette. Comunque, l'intera selezione, che può essere un processo molto lungo, è stata realizzata efficientemente in una quantità di tempo molto piccola usando solo il diskette del blocco indice. Un buon sistema scriverà su un diskette i nomi che stamperà e segnalerà all'utente su un display CRT che in seguito dovrà essere montato un altro specifico diskette. L'utente avrà il tempo per montare il diskette successivo mentre il sistema è occupato a stampare le etichette. Nel momento in cui la stampante, che è un dispositivo lento, finirà di stampare l'ultimo nome del diskette precedente, il successivo sarà già montato. Il sistema funzionerà alla massima velocità meccanica di cui è capace.

Lo svantaggio di dover fornire diskette successivi ad un sistema può essere eliminato comprando dischi più grandi o drivers di diskette a doppio drive e doppia densità. Questo hardware viene descritto nella sezione delle periferiche. Cominciano ad essere disponibili fixed heas disks (dischi a testina fissa) a basso costo.

SOFTWARE FACILITIES ESISTENTI

La maggior parte dei sistemi microcalcolatori disponibili oggi offrono i mezzi hardware richiesti per eseguire tutte le operazioni che sono state descritte a velocità sufficiente, purché non sia richiesta aritmetica complessa. Sfortunatamente, *non sono ancora stati sviluppati programmi commerciali completi, che automatizzino completamente tutte le mansioni richieste.* Si può vedere facilmente dalla descrizio-

ne sopra che il compito del sistema di gestione dei files e dei vari programmi di processing è complesso. Tali programmi sono stati creati solo per i computers più grandi.

Il costo per sviluppare tali programmi è molto maggiore del costo per sviluppare la reale struttura hardware in cui essi risiedono. Per questa ragione, i costruttori non sono di solito ansiosi di sviluppare tali programmi costosi quando si accingono a vendere l'hardware. Ditte di consulenza e software houses tradizionalmente hanno fatto buoni affari vendendo programmi specializzati o packages ad utenti commerciali. Comunque, poiché il software non può essere protetto efficientemente dall'essere copiato, è difficile vendere con profitto programmi software per hardware a basso costo.

La maggior parte dei sistemi a microcomputer disponibili oggi possono offrire pochi programmi commerciali. Di solito essi offrono un sistema di file general purpose, che permette all'utente di creare e manipolare i file simbolicamente. *Questo è totalmente insufficiente.* Devono esistere programmi di sviluppo specializzati, che permettono all'utente commerciale di raccogliere dati automaticamente, comodamente, aggiornarli, e che facciano sì che il computer realizzi gli aggiornamenti in altri file. La maggior parte dei programmi disponibili risolve solo *uno* dei problemi. Di solito, c'è un programma degli acconti ricevibili, ed uno *separato* degli acconti pagabili, ed un programma *separato* d'inventario e così via. Questi "packages" separati sono utili per mantenere files indipendenti. Purché il numero delle transazioni sia alto per ognuno di questi files, essi forniscono un servizio di valore. Comunque, essi fanno solo parte del lavoro. All'utente commerciale viene ancora richiesto di aggiornare manualmente tutti i files che possono essere coinvolti in una singola transazione.

L'IMPIEGO DEI MICROCOMPUTERS PER APPLICAZIONI COMMERCIALI

Ora che sono state messe in evidenza le limitazioni, *è questa una ragione per non considerare un sistema microcalcolatore in un'applicazione commerciale?* No.

Le strutture che sono state descritte sono le strutture ideali per un sistema di automazione commerciale. Comunque, anche soluzioni lontane da questo modello ideale possono offrire benefici molto superiori al costo del sistema. Un sistema efficiente può essere comprato anche con 5 o 10 dollari. Un tale sistema con packages di software minimi permetterà l'automazione dell'inventario, degli acconti pagabili, degli acconti ricevibili e spesso altre funzioni quali le mailing list, back orders, libro paga o calcolo delle tasse. Di solito tali benefici superano di molto in valore il costo iniziale del sistema. Per questa ragione, i sistemi a microcalcolatori sono strumenti di grande valore per un'automazione commerciale limitata. Comunque si dovrebbe tenere in mente che si pagherà un prezzo invisibile. Se vengono creati files costosi, o se vengono scritti programmi specifici, il così detto "investimento soft-

ware" diventerà dominante. Dopo un periodo di tempo, l'investimento per la strutturazione di questi files o per sviluppare nuovi programmi diventerà più significativo del prezzo iniziale d'acquisto del sistema e che il costruttore stia vendendo un nuovo sistema, che non sarà completamente compatibile con il primo. Quindi l'utente commerciale dovrà spostarsi sul nuovo sistema e riformare i propri files e risviluppare nuovi programmi.

Comunque, è una sensazione dell'autore quella che anche se il primo sistema viene essenzialmente abbandonato dopo pochi anni, esso fornirà una transizione estremamente valida all'interno del vero mondo della gestione computerizzata degli affari. È vero che l'investimento hardware iniziale sarà andato perduto dopo pochi anni, e che potrebbe anche andare quanto un consistente investimento software. Tuttavia questo ci sarà tradotto in una strutturazione delle procedure commerciali in operazioni computerizzate, nell'addestramento di personale, e nella consapevolezza delle capacità addizionali che saranno necessarie in sistemi futuri. Tali sistemi forniranno benefici iniziali immediati che di solito supereranno di gran lunga il costo, e forniranno benefici sostanziali all'acquirente. Dopo aver usato un tale sistema per uno o due anni, l'acquirente intelligente che capisce la natura dei suoi fabbisogni, sarà in una posizione adatta per capire completamente il sistema ideale per le sue necessità commerciali. A questo punto la sua seconda scelta è molto probabile che sia ottimale.

Fig. 7-8: Un sistema commerciale (ADC).

È un po' come raccomandare ad un guidatore novello di guidare un'automobile a basso costo prima di comprare l'auto dei suoi sogni. La prima auto può anche essere inadeguata, ma fornisce un valore educativo, espandendo le abilità del suo utente.

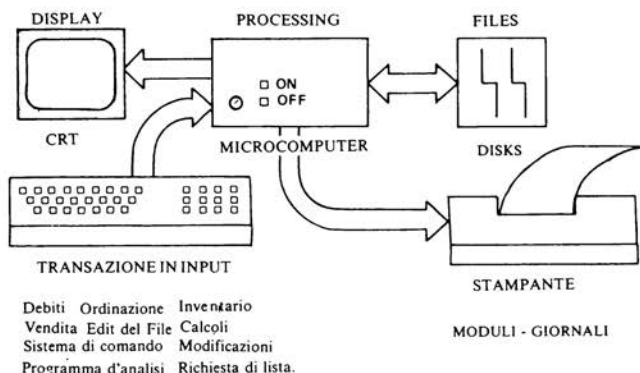



Fig. 7-9: Gli elementi di un sistema commerciale.

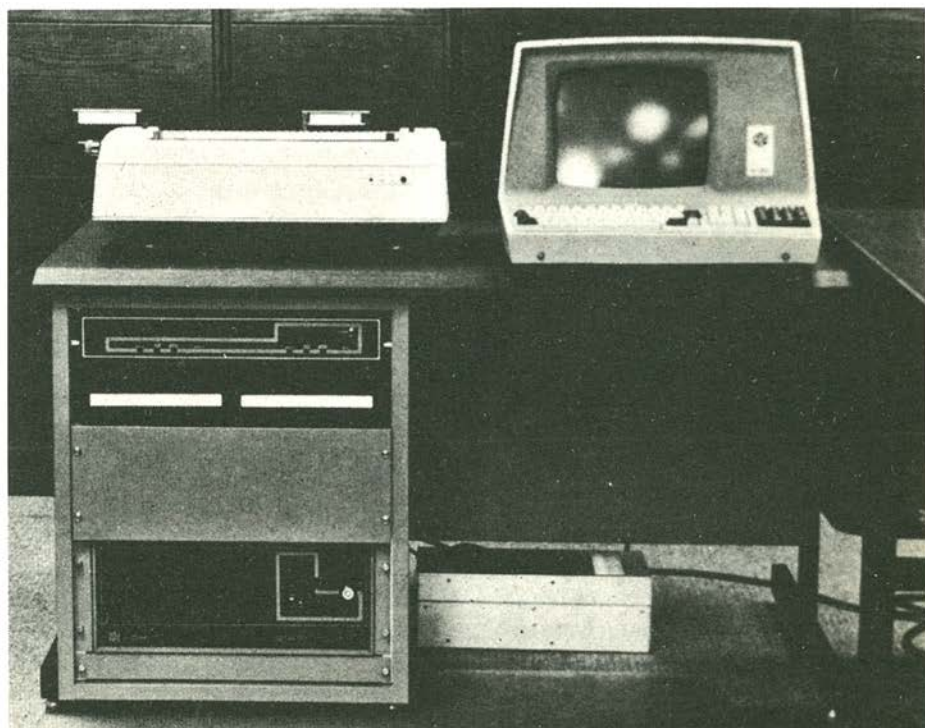



Fig. 7-10: Un altro sistema commerciale "Standard" (Altair).

COSA PENSARE DEI MINICOMPUTERS?

Computers più grandi, più potenti, più costosi potrebbero apparire, a prima vista, in grado di fornire una risposta alla capacità di gestione commerciale per impieghi generali. Questo non è necessariamente vero. Si dovrebbe tenere presente che il 90% dei minicomputers sono usati per applicazioni diverse da quelle commerciali. Solo il 10% circa dei minicomputers sono usati in applicazioni commerciali. La maggior parte dei mini sono usati per automazione industriale ed acquisizione di dati specializzati. La ragione è sempre la stessa: i costruttori vendono hardware. Essi non sono ansiosi di sviluppare packages di software estremamente costosi sui quali essi non possono realizzare un profitto consistente. I migliori sistemi per applicazioni commerciali, cioè quelli che incorporano calcolatori che si vendono per centinaia di migliaia di dollari. Su tali costosi computers con periferiche potenti, diventa ragionevole installare del software altamente costoso.

Tuttavia uno dei parametri dell'equazione è cambiato. La disponibilità di sistemi a microcomputer di costo molto basso significa che, per la prima volta,c'è la possibilità di vendere software che può essere ammortizzato su di un largo numero di unità. Per questa ragione ci si può aspettare che una buona parte dei programmi altamente complessi precedentemente disponibili solo su computer a grande scala diventino disponibili sui microcomputers. Si può prevedere che, entro pochi anni, i microcomputers diventeranno potenti strumenti commerciali equipaggiati con la maggior parte delle funzioni richieste per una gestione commerciale efficiente. I sistemi disponibili oggi sono il primo passo in quella direzione.

PROGRAMMI COMMERCIALI CUSTOM

Una volta che siano state definite le necessità specifiche di una azienda diventa possibile valutare se uno specifico sistema microcalcolatore si adatta a queste necessità. Se non lo fa occorre dotarsi di programmi o packages software addizionali. La domanda essenziale a questo punto è: *vale la pena di scrivere un programma o di appaltarlo a qualcuno esterno per aggiungere programmi specifici richiesti dall'azienda?* La risposta usuale è: *no*. Chiaramente se c'è la possibilità di programmare in casa, o se il valore istruttivo della programmazione è significativo per voi, allora potrebbe valere la pena di farvi da soli i programmi. Il punto importante è che la maggior parte degli utenti commerciali semplicemente *sottovalutano completamente la quantità di sforzi necessari* per fare un programma commerciale utilizzabile, corretto, e documentato. In realtà è possibile sviluppare velocemente un linguaggio ad alto livello come un programma BASIC che sembrerà adattarsi alle necessità principali. Tuttavia, a meno che un tale programma sia sviluppato bene, ben documentato, ben adattato alle necessità dell'azienda, ed al sistema esistente, esso può portare più danno che utile. Si potrebbe scoprire che dopo che un certo numero di articoli sono stati introdotti in un programma di gestione dei file, esso diventa essenzialmente inutile, perché la sua efficienza si disgrega completamente, o sempli-

cemente perché è incapace di manipolarli. Si potrebbe trovare che quando alcuni tipi di dati vengono usati o introdotti si verificano degli errori, che nessuno nella ditta è capace di correggere. È un compito piuttosto lungo e costoso creare un sistema veramente usabile, completo e debugged (corretto dagli errori). A meno che i benefici siano chiaramente giustificati dall'alto costo di svilupppo di un tale programma *di solito la cosa migliore è stare alla larga da qualsiasi sviluppo fatto su misura.* I programmi più appetibili sono quelli che sono già stati sviluppati da un po' di tempo e verificati da altri utenti. In aggiunta, un sistema dovrebbe essere *integrato,* dal punto di vista del software. Tutti i programmi dovrebbero essere in grado di manipolare comuni strutture di files, o di aggiornare i files richiesti. Programmi addizionali semplicemente aggiunti su un sistema potrebbero non essere in grado di fare uso delle sue strutture o potrebbero creare strutture che altri programmi non saranno capaci di usare.

Tuttavia i programmi in packages potrebbero non essere ottimali per la vostra specifica azienda. In questo caso dovrete valutare queste possibilità:

- i programmi in packages possono essere modificati o adottati preferibilmente da un programmatore interno all'azienda. Questo porterà ad una migliore efficienza di programmi per le vostre specifiche procedure o esigenze.
- commissionare, o aggiungere programmi aumenterà il costo totale del sistema, e può diminuire la sua affidabilità.

SOMMARIO

La maggior parte dei sistemi a microcomputer sono disponibili attualmente con dei semplici packages commerciali. Questi programmi sono progettati per *automatizzare le transazioni.* Qualsiasi azienda con un grande numero di transazioni *ripetitive* otterrà benefici dall'automazione computerizzata.

Tuttavia, questo è vero solo se il tipo di "package standard", disponibile insieme al sistema, si adatta all'azienda. Per esempio, un programma di annotazione delle vendite potrebbe formulare tante domande da richiedere il doppio del tempo necessario per uno sviluppo manuale (questo è usuale). In tal caso, l'automazione è conveniente solo se ne derivano vantaggi addizionali. Tali benefici addizionali sono: miglioramenti nella procedura di introduzione dei dati, processing o aggiornamento di altri file automatico, generazione di report.

Se questi programmi addizionali o benefici esistono, allora persino un programma di introduzione dati piuttosto "standardizzato", non ottimale, ingombrante, porterà benefici.

Questa è sempre l'assunzione di fondo. **Avvertimento**: se queste strutture addizionali non esistono ancora nel sistema che state considerando, dovreste valutare con cura le varie possibilità. Molte piccole aziende trovano il processing manual degli ordini o la contabilità manuale più pratica, ed economica di un package commerciale "standard".

Se è disponibile un programmatore part-time, o può essere giustificato possono essere sviluppati molti programmi special purpose che si adattano perfettamente all'azienda. Se il programmatore è abile, si otterranno tutti i benefici potenziali di un sistema a computer, che daranno come risultato un certo numero dei benefici descritti sopra:

- efficienza, costo più basso, affidabilità aumentata, informazione di management istantanea, raccolta di ricevibili migliorata, automazione delle nuove procedure commerciali (ri-ordinazione automatico - verifica del credito, spedizione postali specializzate).

I microcomputer disponibili oggi offrono tutte le risorse hardware necessarie per meno di 10.000 dollari. Il software è la chiave.

ACQUISTO DEL SISTEMA-UN SOMMARIO

1 - Ci deve essere una necessità

La natura e la quantità di lavoro devono giustificare il cambiamento. Questo può essere misurato tramite il numero di transazioni identiche o reports che devono essere generati. In aggiunta, il sistema a computer può essere giustificato dagli unici problemi, che può risolvere in situazioni speciali, come la gestione d'inventario, lista di spedizione postale, rapporti di management.

2 - Quando comperare?

Il prezzo per i componenti elettronici continua a scendere ogni anno mentre i prezzi per le periferiche tendono a rimanere stabili o a diminuire lentamente.

Il sistema di domani sarà sempre più economico di quello di oggi.

Tuttavia, un sistema a computer significa il risparmio di N dollari per mese. Posticipare la sua installazione di m mesi equivale perdere $m \times N$ dollari.

3 - Quale comperare?

Tutte le opzioni principali ed equipaggiamenti disponibili saranno presentati ora.

Altre Precauzioni Standard

- Create un back-up (supporto) sul disk alla fine di ogni giornata, così che qualsiasi file danneggiato o perduto possa essere ricreato.
- Proteggetevi dalle fluttuazioni dell'alimentazione se la vostra area è affetta da tale problema.

SOMMARIO

I microcomputers sono stati classificati in tre categorie: a scheda singola e general (per la didattica), sistema domestico (giochi più processing limitato) e general purpose. Entro la categoria dei microcomputers general purpose, esiste attualmente un vasto numero di prodotti che offrono varie combinazioni di caratteristiche già incorporate nel dispositivo.

Per l'utente commerciale: sono adatti solo quei microcomputers che offrono:

- piena memoria (fino a 64K)
- BASIC completo, residente in ROM (deve risiedere nella ROM per l'efficienza).
- insieme completo di periferiche, o un bus standard a cui possano essere collegate altre periferiche.
- software completo, compreso un DOS (Disk Operating System) ed un sistema di files.
- disponibilità di package commerciali che possano essere eseguiti direttamente su questo processore.

Per uso personale: la maggior parte dei sistemi offrono prestazioni simili e opzioni, I/O differenti. L'analisi di cui sopra dovrebbe essere d'aiuto nel fornire dati di confronto.

STRUTTURE COMMERCIALI DESIDERABILI
UNA LISTA DI VERIFICA

1 - LISTE

- ☐ Clienti
- ☐ Fornitori
- ☐ Inventario
- ☐ Ordinazioni
- ☐ Venditori
- ☐ Lista di spedizioni
- ☐ Lista di prelevamento
- ☐ Impiegati

2 - REGISTRI

- ☐ Pagamenti ricevuti
- ☐ Registro generale
- ☐ Acconti ricevibili
- ☐ Acconti pagabili
- ☐ Acconti bancari
- ☐ Registrazioni fiscali

3 - MODULI COMMERCIALI

- ☐ Bollette
- ☐ Etichette
- ☐ Assegni
- ☐ Libro paga
- ☐ Lettere circolari
- ☐ Rendiconti
- ☐ Avvisi speciali
- ☐ Resoconti fiscali

4 - RESOCONTI

- ☐ Acconti dovuti in passato
- ☐ Resoconti delle vendite
- ☐ Resoconti d'inventario (disposti per categorie, per articolo, per età dell'articolo ecc.)
- ☐ Back orders (ordinazioni arretrate)
- ☐ Acconti disposti cronologicamente (ricevibili, pagabili)
- ☐ Mailing list classificate
- ☐ Resoconto delle vendite ai clienti
- ☐ Tutti i file immagazzinati mediante un criterio specifico.
- ☐ Commissioni e adempimento dei venditor
- ☐ Straordinari, vacanze, assenze di malattia degli impiegati.

CAPITOLO 8

SCEGLIERE UN SISTEMA

INTRODUZIONE

La scelta di un sistema microcalcolatore può essere guidata dagli stessi principi generali della scelta di un sistema altamente complesso. Esistono numerose possibilità che si traducono in una vasta gamma di vantaggi e svantaggi in prestazioni e convenienza. Per effettuare una scelta ragionata l'utente dovrebbe per prima cosa definire (e comprendere) i propri obiettivi principali. Una volta che sono stabiliti gli obiettivi essenziali, possono essere usati criteri di selezione razionali. Poiché la maggior parte degli utenti, in realtà, si aspetta che il proprio sistema sia il migliore in tutti i sensi (e naturalmente economico) potrebbero verificarsi grosse delusioni. Nella prima parte di questo capitolo saranno presentati i criteri di valutazione usuali. Questa sarà un'analisi dei criteri possibili. Nella seconda parte saranno presentati in una sintesi i tipi di sistemi più adatti per particolari classi di applicazioni. Infine, nel Capitolo 10 sarà presentato un panorama dei principali sistemi esistenti in commercio.

CRITERI DI SCELTA

I criteri di scelta principali sono di solito: basso costo, prestazioni, hardware completo, software completo, comodità ed affidabilità. Soffermiamoci su ognuno di questi criteri.

Basso costo

Da un punto di vista generale, la competizione di prezzo tra i vari costruttori è talmente dura che in pratica tutti i sistemi equivalenti vengono venduti ad un prezzo simile, per la stessa configurazione. Perciò il basso costo non può essere adoperato come criterio per qualsiasi costruttore specifico, ma tende generalmente a rappresentare un più piccolo insieme di componenti o strutture per un dato sistema ad un dato tempo. Naturalmente, le guerre sul prezzo possono rendere in qualsiasi momento un particolare sistema più attraente di un altro per un periodo di tempo limitato.

Come regola generale, l'utente motivato dal più basso costo possibile può per prima cosa dare un'occhiata ad un single-board (computer a scheda singola). Egli potrà comprare un sistema microprocessore minimo per meno di 250 dollari, che

richiederà un'alimentazione extra che costa 50 dollari. Ora egli potrà comunicare in esadecimale. Se le ambizioni dell'utente sono un po' alte, ed egli intende fare della vera programmazione, egli avrà bisogno di comprare un sistema che abbia come minimo: un PC board (Printed Circuit board Circuito stampato) con almeno 4K di RAM, un contenitore, un'alimentazione, una tastiera alfanumerica, un display CRT per l'uscita ed un mangiacassette come mezzo di immagazzinamento di massa.

Comunque, si deve sottolineare che se l'utente ha in mente un sistema più completo, il costo della CPU in sé non è più il costo dominante nell'intero sistema. *Il costo principale di un sistema sono le periferiche.* Una buona stampante da sé sarà più costosa della CPU. In aggiunta l'intero costo del sistema è di solito per il software una risorsa costosa in qualsiasi sistema.

Per questa ragione, si può sostenere che, eccetto che per il caso di una spesa veramente minima, indipendente da altri criteri, il costo effettivo della scatola microcomputer non è un criterio di selezione importante.

Evoluzione del prezzo

Ogni anno i prezzi diminuiranno, in pratica, per tutti gli elementi elettronici di un sistema. Questo è sempre stato vero nella storia dei computers, e non è ancora stato raggiunto il limite. Comunque, il costo delle periferiche può non diminuire significativamente.

Dopo che si è fatta una scelta, e che il sistema è stato consegnato, ci sarà una opzione di costo più basso per almeno uno degli elementi del sistema!

Questo non significa che uno dovrebbe soltanto aspettare. Fate la scelta migliore ora, e iniziate ad usare il sistema. Il divertimento nel caso di un personal computer o i benefici nel caso di un sistema commerciale incominciano ad accumularsi immediatamente, e sono molto più grandi di qualsiasi risparmio ottenuto col non fare nulla.

Per di più il guadagno in conoscenza che voi otterrete usando un sistema a computer porterà grossi benefici sia nella vostra vita professionale, sia nella vostra scelta di sistemi futuri.

In breve, una volta che si è stabilito che il livello di prezzo globale è meritevole, non perdete tempo per risparmiare il 10 o 20 per cento.

Prestazioni

La cosiddetta "performance" può essere completamente ingannevole. Per prima cosa osserviamo le prestazioni dell'hardware. Le *prestazioni dell'hardware* di un sistema sono spesso misurate con l'alta velocità della CPU a cui si deve accoppiare un tempo di accesso basso della memoria (è sufficiente non avere mai la CPU che aspetta dati).

Per prima cosa, la velocità della CPU tende ad essere pubblicizzata dal clock rate

(frequenza del clock) "questo microcomputer ha un clock rata di 2MHz (megahertz)". "Quest'altro ha un rate di 4MHz". Significa che il secondo processore è due volte più veloce? No, non necessariamente. Semplicemente non c'è nessuna relazione ovvia tra il clock rate e la reale velocità alla quale vengono eseguite le istruzioni. Il clock rate è il ritmo al quale gli impulsi sono trasmessi al chip della CPU così che esso possa propriamente eseguire le istruzioni. L'esecuzione di una istruzione richiederà diversi periodi di clock. Non ci saranno nemmeno due microprocessori che useranno lo stesso numero di impulsi per le loro istruzioni. Per questa ragione, il clock rate di diversi microprocessori è una misura ingannevole del rendimento. Naturalmente, confrontare due clock rate differenti dello stesso microprocessore, per esempio uno Z80 a 2MHz con uno a 4MHz, è invece rilevante. Comunque questo non può nemmeno significare che lo Z80 a 4MHz sarà il 50% più veloce dello Z80 a 2MHz. Questo perché i dati e le istruzioni devono essere presi dalla memoria. Questo implica che tutti gli accessi alla memoria devono essere il 50% più veloci, il che talvolta non è la realtà (la velocità della memoria deve accoppiarsi con la velocità del processore).

Osserviamo ora la *performance della memoria.* Le prestazioni della memoria sono di solito caratterizzate dal tempo di accesso della sua RAM, cioé il tempo che ci vuole per prendere una parola dalla RAM. Più la RAM è veloce e più è costosa. Poiché la tecnologia della memoria RAM progredisce costantemente, oggigiorno è difficile stabilire una velocità tipica per una RAM, perché è in pratica assicurato che entro un'anno ce ne sarà una più veloce per lo stesso prezzo, o per un prezzo inferiore. A condizione che la RAM sia abbastanza veloce, così che le istruzioni o i dati possono essere mandati al microprocessore senza ritardo, non si ottiene alcun vantaggio nell'usare una memoria più veloce. Uno deve semplicemente fare sì che la velocità della memoria sia sufficiente per la velocità del microprocessore, quando valuta il rendimento del sistema. È interessante notare che alcuni sistemi di basso-costo usano una memoria lenta, che rallenta l'esecuzione di un processore veloce. Supporremo che la maggior parte dei sistemi, che saranno considerati, abbiano una memoria la cui velocità sia sufficiente per il microprocessore considerato. In tal caso la cosiddetta performance della memoria è irrilevante per la scelta. Abbiamo appena perso un criterio di scelta. Come faremo a valutare le prestazioni dell'hardware?

Benchmarks

Devono essere usati i benchmarks. L'unico vero modo per valutare il rendimento di una CPU hardware è di far eseguire *benchmarks programs.* I benchmarks programs sono programmi qualificati come "tipici dell'applicazione considerata". Un benchmark deve essere fatto operare su diversi computer e poi possono essere comparati i tempi calcolati. Non esiste una cosa come un'"istruzione tipica" per un computer, e, per questa ragione il confronto di istruzioni specifiche tra di loro può essere completamente ingannevole. Sfortunatamente non c'è un benchmark pro-

gram standard. Per applicazioni specifiche come il trasferimento di dati paralleli, per esempio, può essere scritto un programma di trasferimento dati paralleli, e possono essere stabiliti i benchmarks (livelli di velocità). Nella maggior parte dei casi, il campo di applicazione che deve essere eseguito è talmente grande, che è molto difficile scrivere un ragionevole benchmark. Per di più un benchmark program, scritto dal costruttore è quasi inutile. *Per essere valido, un benchmark program deve essere scritto dall'utente del sistema,* poiché egli è quello che programmerà il sistema. Il costruttore può introdurre miglioramenti intelligenti, spendendo mesi, o anni su esso, così che il suo benchmark program verrà eseguito molto più rapidamente sul suo processore, che sui processori dei concorrenti.

Comunque, al di fuori di alcuni programmi per applicazioni specifiche, non c'è una cosa come un "programma medio'. Alcuni programmi calcolano pesantemente, altri eseguono l'input-output. Perciò, non c'è un "benchmark standard".

Se si deve valutare un sistema "general purpose", i programmi possono solo essere usati come un approssimativo criterio di valutazione del rendimento del sistema. Abbiamo quasi perso il nostro terzo cosiddetto "criterio della migliore selezione". Cosa possiamo usare?

I criteri importanti

Per prima cosa la reale velocità hardware della MPU non è il criterio di scelta più significativo per la performance globale del sistema.

Secondo, la maggior parte dei microprocessori sul mercato oggigiorno hanno in pratica le stesse prestazioni. Si può sostenere che i più veloci sono spesso due volte più veloci dei loro competitori principali. Comunque, questo sarà importante *solo se il software del sistema è efficiente. L'unico criterio veramente importante per il funzionamento di un sistema, se deve essere programmato in linguaggio ad alto livello, è l'efficienza del software.* Anche se il sistema deve essere programmato in linguaggio a livello assembly, l'abilità del programmatore può avere un impatto più significativo sulla velocità globale del sistema che non il rendimento effettivo dell'hardware. Per di più è probabile che entro uno o due anni, saranno disponibili chip più veloci così che miglioramenti in velocità possano essere ottenuti ad un costo minimo sostituendo uno o più boards. Come al solito, la velocità del software è la chiave delle prestazioni del sistema.

Infine, il grave ostacolo alla velocità di un sistema microcalcolatore usato per applicazioni personali e commerciali non è al livello del processing. È a livello di input-output. Servono soprattutto un disk veloce ed una stampante veloce, i quali sono entrambi più costosi dell'intera scatola del microcomputer.

Le periferiche di solito rappresentano il fattore che limita il rendimento per il processing orientato sui file (applicazioni commerciali).

La maggior parte delle applicazioni personali e commerciali sono scritte in un linguaggio ad alto livello, come il BASIC.

Nel caso di un linguaggio ad alto livello, l'efficienza di un interprete (o compilatore) è decisiva.

Come esempio alcuni degli interpreti peggiori di BASIC sono *50 volte più lenti* dei migliori. Finché si confrontano dei microprocessori tra di loro, è usuale un rapporto di *1 a 2*. Quando si confronta il software, non è inusuale avere un rapporto di 1 a 50.

Chi ha il software migliore? Sfortunatamente, la risposta a questa domanda varia rapidamente. È corretto dire che nel momento in cui scrivo, un certo numero di costruttori hanno ovviamente interpreti migliori di altri. Tuttavia, gli interpreti vengono continuamente migliorati, così che la situazione può cambiare rapidamente. Qualsiasi classificazione qui sarebbe ingannevole. Regolarmente, riviste per hobbisti (elencate alla fine di questo libro) pubblicano confronti fra vari interpreti, ed altri programmi, usando benchmarks. Il lettore interessato è rimandato alla lettura di queste misure.

Semplicemente, il lettore che sta esaminando la performance di un sistema è vivamente avvertito di esaminare la velocità dei mezzi software che intende usare.

Hardware completo

Un sistema base consiste di:

- scatola del microcomputer, con possibilità di espansione della memoria, e slots per interfacce periferiche addizionali, con un alimentatore sufficiente per un funzionamento affidabile di una scatola completa.
- tastiera d'input di configurazione adeguata
-output CRT
-output per hard copy (stampante)
-memoria di massa - come minima cassetta, preferibilmente disk.

Se il sistema usa il bus S100 standard industriale, la completezza dell'hardware non importa, poichè tutti gli elementi possono essere interscambiabili fra i costruttori.

In caso contrario, è cruciale la completa disponibilità di tutti gli elementi specialmente se si desidera un'espansione futura. S100 e altri bus saranno esaminati in seguito.

L'acquirente di computer prudente dovrebbe sempre assumere che non saranno mai disponibili altre periferiche od opzione oltre a quelle già esistenti per davvero. "Per davvero" significa che un sistema esposto ad una mostra può essere "uno di una varietà".

Infine, potrebbero esistere speciali esigenze come la disponibilità di un floating point board o di un dispositivo d'interfaccia specifico.

Le caratteristiche di tutte le periferiche usate in un sistema saranno presentate e valutate, in dettaglio, nel prossimo capitolo.

Software completo

Un sistema, anche se meraviglioso è quasi inutile a meno che sia equipaggiato con tutte le strutture software richieste dalla vostra applicazione.

Esso deve includere un monitor, un editor, ed un processore di linguaggio come un interprete BASIC (per maggiori dettagli guardate il capitolo sulla programmazione).

È molto costoso sviluppare software, e molti costruttori cercano di fornire il minimo con cui riescono a cavarsela.

Nel caso di applicazioni commerciali devono essere applicati i seguenti criteri:

1 - disponibilità di package pronti all'uso da far funzionare sul vostro sistema, qualsiasi sia il linguaggio.
2 - un BASIC completo (per dettagli maggiori guardate il capitolo sul BASIC)
3 - un buon sistema di file, per una manipolazione automatica dei vostri file.

Per un sistema didattico

1 - disponibilità di un BASIC completo
2 - disponibilità di un software time sharing (time sharing = l'utilizzazione ottimale di un computer e dei suoi dispositivi periferici con cui si tiene conto del tempo di processing differente per ogni macchina, e si opera di conseguenza) per un accesso multiplo simultaneo al sistema di vari utenti.

Per un sistema personale

1 - un BASIC completo
2 - un buon sistema di file
3 - un assembler

Quest'ultimo punto potrebbe sembrare ovvio. Tuttavia, poiché nel momento in cui scrivo alcuni "personal computers" sono equipaggiati con un BASIC accettabile, ma non con un assembler accettabile, e neppure con una connessione tra essi, precludendo così l'uso conveniente delle routine in linguaggio assembly.

In aggiunta, ed in tutti i casi, ci dovrebbe essere un editor completo e potente per un editing comodo (per i dettagli guardate il capitolo sulla programmazione).

Convenienza

La convenienza è una delle chiavi per l'utilizzazione riuscita del sistema. Alla maggior parte degli utenti non interessa reinventare la ruota, e nemmeno dover creare interfacce complesse, o persino riprogrammare algoritmi, tutte cose che sono già state inventate molto tempo fa. Per questa ragione, i microprocessori sono i più usati tra i microcomputer personali. La maggior parte dei microcomputer usano l'8080 della Intel, lo Z80 della Zilog, il 6800 della Motorola, ed il 6502 della MOS Technology. Caso interessante la Zilog è stata una ditta derivata dalla Intel. A causa dell'abbondanza di software disponibile per l'8080 o lo Z80 (essi hanno un insieme di istruzioni in comune) come pure per il 6800 della Motorola ed in misura minore per il 6502 della MOS Technology, questi microprocessori sono ora quelli più largamente usati per i microcomputer personali.

Convenienza significa che sono disponibili tutte le funzioni richieste per far sì

che il sistema possa usarle immediatamente, e che lo sforzo ed il tempo richiesti siano minimi.

In particolare convenienza significa:

1 - un sistema completo pronto all'uso da un punto di vista hardware e software
2 - facilità di gestione. Questo implica per lo più aiuti software potenti (editor, interprete, sistema di file), ed hardware non aggravante (disk veloce, stampante veloce)
3 - disponibilità e utilizzabilità immediate
4 - documentazione eccellente, spesso un fattore chiave per l'uso efficiente del vostro tempo.

Rendimento del sistema globale

Se è probabile che il sistema a computer manipoli file, piuttosto che eseguire programmi che elaborino dati residenti per intero nella memoria centrale del sistema, l'efficienza del dispositivo di immagazzinamento di massa è essenziale per il rendimento del sistema. In particolare per le applicazioni commerciali, i file saranno vasti e non risiederanno simultaneamente nella limitata memoria centrale del sistema limitato a 40 o 48K (16K sono di solito consumati dal programma nella ROM). La velocità del disk, sia un floppy disk che un hard disk diventa decisiva.Infatti il tempo di accesso del disk è più importante che non la velocità di esecuzione del microprocessore nei casi in cui è richiesto un accesso ripetuto ai file (specialmente in applicazioni commerciali). Se c'è una grande quantità di file che devono essere stampati, diventerà necessaria una stampante ad alta velocità, altrimenti il funzionamento del sistema verrà rallentanto. Una stampante ad alta velocità è di solito la periferica più costosa. Per questa ragione, vengono invece usate stampanti a basso costo, il che porta ad una globale inefficienza del sistema, e l'utente deve aspettare che il file sia completamente stampato prima di poter proseguire il proprio lavoro.

Affidabililtà

La maggior parte dei sistemi microcalcolatori, quando furono progettati per la prima volta, non furono costruiti agli stessi standard di affidabilità dei precedenti minicomputer. Essi furono costruiti con entusiasmo, da un solo ingegnere, trovarono accidentalmente un mercato e non poterono più essere migliorati poiché la loro configurazione fu infatti standardizzata a causa del numero molto alto di vendite. Tuttavia, l'alta integrazione intrinseca alla circuiteria LSI è tale che il numero di componenti è molto minore che nei minicomputer tradizionali, nel passato si hanno come risultato sistemi la cui affidabilità globale è paragonabile a quella del minicomputer tradizionale. Per tutti gli scopi pratici la maggior parte dei microcomputer, sono altamente affidabili in un ambiente "normale" dopo che essi hanno lavorato correttamente le prime 200 ore. Questa è chiamata la fase di "burn-in" (stagionatura o rodaggio) durante la quale si verifica la maggior parte dei guasti. La probabilità di avarie future è molto minore. Un ambiente "normale" significa che non ci siano variazioni drastiche di temperatura e nemmeno della percentuale

di umidità. Talvolta si incontrano anche problemi di "rumore" usando un corredo di bus ed una distribuzione dell'alimentazione scarsi quando vengono aggiunti boards addizionali.

Il principale pericolo deriva dal successo dei sistemi a microcomputers. Per continuare a fornire sistemi ad un costo sempre più basso, alcuni costruttori si dirigono occasionalmente a fornitori di componenti meno affidabili. È possibile acquistare a prezzi molto più bassi nel mercato "grigio" componenti che non sono stati *controllati*. Questo andrebbe bene purché i componenti venissero provati al loro arrivo. Però quando il tempo preme, questo può non avvenire, così che componenti non provati e potenzialmente inaffidabili possono essere immessi in un sistema. Il fatto grave è che i guasti possono accadere durante un evento relativamente poco frequente, come ad una bassa temperatura, alta temperatura o semplicemente attraverso una deteriorizzazione progressiva del componente, e questa può avvenire dopo che la garanzia è scaduta. Di solito, il costruttore rimpiazzerà lo stesso il componente difettoso. Comunque, l'individuazione del componente difettoso può essere un consumo di tempo ed un compito frustrante. L'affidabilità del costruttore, e la sua reputazione, è perciò un criterio di scelta, importante per cautelarsi da questo pericolo.

TIPI DI SISTEMI

Kit, board o sistema?

I microcomputer esistono in due forme principali: microcomputer a board singolo e sistemi completi impacchettati in contenitori. In aggiunta c'è di solito l'opzione di un sistema completamente assemblato o di un kit. Qual'è quello da scegliere?

I vantaggi e le limitazioni di ogni scelta sono semplici: un board ha il vantaggio di un costo minimo e lo svantaggio di risorse hardware limitate. Una scatola offre le capacità hardware di un minicomputer tradizionale, ma costa di più. Esaminiamo ora in dettaglio le alternative.

Microcomputer su singola scheda

Una scheda tipica include l'unità microprocessore più qualsiasi chip di controllo richiesto, una quantità di memoria limitata come 512 parole di ROM, e da 2K a 4K di RAM, ed un numero minimo di interfaccia come un'interfaccia per telescrivente, un'interfaccia per mangiacassette, e dei port (connessioni di input o output) input/output general purpose.

A causa della sua struttura standardizzata, un board disponibile deve essere un compromesso tra le varie strutture che devono risiedere su un board. Queste strutture sono la quantità di memoria, di I/O, di interfaccia, e la presenza di una tastiera. Sotto, sarà mostrato che la limitazione di memoria è di solito la più grave, poiché preclude l'uso di un'assembler o l'uso di un linguaggio ad alto livello.

Prendere la scheda giusta può essere difficile.

Considerando queste limitazioni, un board può essere usato di per se stesso essenzialmente per due scopi.

1-Come dispositivo di controllo. Un tale board fornisce la potenza di elaborazione di un computer completo ad un costo di una o poche centinaia di dollari. Può essere usato per consentire il controllo diretto di relé, motori, o altri dispositivi.
È un dispositivo di controllo a basso costo, programmabile, general purpose, che può essere usato in numerosi modi. La sua principale applicazione è nell'automazione industriale, o possibilmente in un ambiente casalingo o di ufficio, per controllare applicazioni semplici, come la regolazione di motori, controllo di processi, controllo dell'ambiente, allarmi per scassinatori temporizzazione automatica o spruzzatori d'acqua (anti incendio).
2 - Come strumento didattico. Un board ha un valore importante per comprendere le interconnessioni di un sistema microcalcolatore tipico come pure per imparare a programmare in linguaggio macchina. In questo ruolo, *esso è tuttavia, essenzialmente un giocattolo.* Non è ragionevolmente possibile sviluppare programmi lunghi o complessi su di un board singolo. A causa della quantità di memoria limitata in-

clusa nella scheda, essa è di solito incapace di far operare un programma *assembler* o un programma *interprete*. Per questa ragione, è necessario, sulla maggior parte dei boards, introdurre istruzioni attraverso la tastiera sotto forma di una sequenza di numeri esadecimali. Questo è un processo altamente incline agli errori e molto lungo. La maggior parte degli utenti saranno capaci di introdurre in questo modo forse poche dozzine di istruzioni, e di imparare così tutte le basi della programmazione a livello macchina. Se viene inclusa una quantità sufficiente di memoria (di solito sotto forma di board esterno) può essere possibile eseguire un programma assembler sul board. Questo permette l'uso di programmi *simbolici*. Tuttavia, lo scrivere programmi simbolici richiede in compenso l'uso di una *tastiera alfanumerica* o di un dispositivo di input/output standard come una telescrivente. Se si intendono spendere da 1000 a 1500 dollari su una teletype, si può sostenere ragionevolmente che in primo luogo ci potrebbe essere una scelta migliore che non un board singolo.

Per riassumere, un board è un dispositivo di computing molto economico che può essere usato efficientemente come un controllore dedicato ad usi particolari, o anche come uno strumento educativo limitato per comprendere la struttura di un microcomputer e per imparare la programmazione elementare. Come esempio finale, un board singolo può essere usato in una macchina commerciale ma non come sistema commerciale.

Il sistema a microcomputer

Il computer, in sé, consiste di uno o più boards cablati in una scatola e integrato con una quantità minima di periferiche richieste per effettuare un uso ragionevole del sistema. Sistemi completi non sono disponibili per appena 500 dollari e per appena il doppio del prezzo di un board singolo. Per questa ragione, essi offrono molte alternative attraenti in più per un uso personale. *Un sistema tipico* include la scatola del computer più una *tastiera alfanumerica, un display CRT, ed un dispositivo di memoria di massa.* I sistemi più economici forniscono un *nastro a cassetta* come memoria di massa mentre quelli più potenti forniscono *floppy disk drivers*. In aggiunta, per qualsiasi applicazione pratica, è richiesta una stampante,

In teoria, un sistema microcomputer può fare qualsiasi cosa che può fare un computer, entro le proprie limitazioni hardware e software. Il suo svantaggio essenziale è di essere meno potente di un minicomputer tradizionale, cioé più lento e con una memoria e delle capacità di input/output più ristrette. Si possono differenziare due tipi di sistemi: sistemi di costo minimo e microcomputer general purpose.

Il sistema di costo minimo

I sistemi di costo minimo comprendono la scatola del microcomputer con 4K parole di memoria, il contenitore, l'alimentatore, la tastiera, un monitor CRT ed un mangiacassette.

Ancora, questa configurazione ha lo stesso potere di processing di qualsiasi altro microcomputer, ma è limitata dalla sua memoria e dalle sue periferiche. La memoria di solito può essere ampliata e spesso sono disponibili periferiche alternative. Tuttavia, solitamente non c'è a disposizione un "software continuo". Il programma (del *sistema di funzionamento*), necessario a supportare le varie periferiche e dimensioni di memoria deve essere progettato correttamente fin dall'inizio. Un sistema minimo è progettato proprio come tale. Esso include un sistema di funzionamento minimo chiamato monitor, che manipola la tastiera ed il mangiacassette. Per trasferirsi su una configurazione più ampia è necessario un sistema di funzionamento differente.

Il sistema a costo minimo offre un programma assembler, ed un mini-BASIC, od un altro programma interprete. L'utente ora può scrivere programmi simbolici a livello assembly o programmi scritti in BASIC (semplificato). In aggiunta egli può immagazzinare o recuperare i propri programmi da un mangiacassette, una comodità importante.

Un simile sistema normalmente è fornito di nastri a cassetta che contengono una varietà di programmi come giochi o programmi "utili": gestione del libretto degli assegni, istruzione matematica, ecc.

Per di più, questo sistema è limitato dalla sua memoria e dalle sue periferiche. Può essere usato per scopi personali, ma non come sistema commerciale.

Questo mostra ancora una volta un fatto che è stato fondamentale per tutti i sistemi a computer: il costo della unità di processo in sé è una piccola frazione del costo totale di un sistema adoperabile. Il costo principale dell'hardware è, ed è sempre stato, il costo delle periferiche.

Come commento finale, il board del microcomputer ha bisogno di non essere collocato in un contenitore grosso ed è spesso contenuto nel contenitore della tastiera stessa, o persino nel contenitore del CRT.

In sommario, sistemi di costo minimo sono piacevoli ed istruttivi giocattoli con i quali si possono fare giochi, eseguire un certo numero di programmi educativi, o imparare a programmare. Tuttavia, poiché essi sono equipaggiati con una piccola quantità di memoria e con un dispositivo di immagazzinamento di massa lento, la cassetta, essi non sono utilizzabili come sistemi commerciali poiché essi non possono fornire una manipolazione efficiente dei file. Essi possono in realtà essere usati per un grande numero di applicazioni personali. A condizione che venga acquistata una memoria addizionale essi possono eseguire un assembler, un editor, o un programma interprete così da sostenere la maggior parte delle applicazioni. Comunque bisogna notare che alcuni di questi sistemi offrono la capacità di elevarsi al livello di sistemi microcomputer general purpose, che è una considerazione importante nella scelta di un sistema.

Il microcomputer general purpose

Il microcomputer general purpose è l'equivalente funzionale di ciò che era il minicomputer. È un box contenente un board processore più i boards addizionali ne-

cessari per l'interfaccia della memoria e di input-output. Il contenitore è parzialmente vuoto e permette l'inserimento di boards addizionali così che la memoria può essere ampliata fino a 40K o 48K e così che vari boards addizionali possano essere inseriti come collegamento con una certa quantità di dispositivi con queste periferiche: uno o preferibilmente due floppy disk drivers una tastiera alfanumerica di buona qualità, ed un display CRT di buona qualità. In più bisogna aggiungere al sistema una stampante adatta. Tali microcomputers general purpose dovrebbero anche includere un alimentatore *capace di alimentare il massimo numero di schede con cui esso può essere equipaggiato.* Un sistema di questo tipo non è più limitato dai difetti precedenti ed è essenzialmente capace di eseguire qualsiasi tipo di programma che non affatichi le sue capacità di processo.

In pratica, il microcomputer può essere inserito in una varietà di contenitori, o persino nel contenitore della tastiera o del CRT. Per le flessibilità nell'aggiungere i boards addizionali, è di solito inserito in un contenitore a parte, delle dimensioni di un sintonizzatore Hi-fi.

I suoi vantaggi rispetto ad un minicomputer tradizionale sono:

- costo più basso (forse 10 volte meno)
- dimensioni minori
- minore consumo d'energia e minore dissipazione di potenza: come risultato l'alimentatore è più piccolo e più economico e servono poche ventole per la ventilazione (per raffreddare i componenti). Questo comporta anche la possibilità di un contenitore elegante, in uno stile vicino a quello dell'equipaggiamento Hi-fi domestico.
- disponibilità di periferiche nuove a basso costo

Le sue limitazioni principali a confronto di un minicomputer sono:

- minor potere di processing.

L'uso di un microprocessore a 8 bit dà come risultato una velocità di esecuzione minore. Tuttavia, eccetto nel caso di calcoli aritmetici, questa limitazione di solito non si fa sentire. Il basso costo del microcomputer ne permette l'uso per un compito specifico, eliminando la necessità di gestione di compiti complessivi o di time sharing software. Il potere di processing di un microcomputer è sufficiente per la maggior parte delle applicazioni personali e commerciali.

- meno software

Poiché i microcomputers sono più recenti la loro libreria di software è inferiore a quella dei minicomputers tradizionali. Tuttavia, il divario viene ridotto costantemente, e sarà presto disponibile una grossa quantità di software.

- minori capacità I/O

Un microcomputer ha una fascia input-output limitata, paragonato ad un mini, Tuttavia, questo è ancora un inconveniente piccolo per la maggior parte degli usi.

In sommario, un microcomputer offre la maggior parte delle caratteristiche dei precedenti minicomputers eccetto che per un processo numerico lento, ad una frazione del costo dei mini. Non eliminera i minicomputers, poiché ha semplicemente creato un nuovo mercato, dove è possibile dedicare un microcomputer ad un com-

pito specifico. Comunque per tutti gli scopi pratici, esso porta il potere di elaborazione dati di un "mini" a qualunque utente.

Pannello frontale o nessun pannello frontale?

Alcuni contenitori di microcomputer sono equipaggiati con un pannello frontale ed alcuni no. Un pannello frontale è semplicemente una fila di luci ed una fila d'interruttori, più luci, interruttori e pulsanti misti. Un pannello frontale è destinato a facilitare l'hands-on-debugging (debug a portata di mano, manuale) quando si programma in linguaggio assembly. Esso visualizza il contenuto binario dei bus e dei registri. Per applicazioni di controllo, che saranno di solito programmate in linguaggio a livello assembly, la presenza di un pannello frontale è favorita dalla maggior parte dei programmatori che abbiano comprensioni dell'hardware, poiché esso visualizza istantaneamente la condizione dei registri o dei bus senza la necessità di una sequenza di ordine per ottenere le informazioni. Comunque il costo del pannello frontale è alto, poiché esso richiede un board di controllo separato più un programma speciale entro il sistema. Esso aggiunge forse cento dollari al prezzo. Per questa ragione, oggigiorno la maggioranza dei sistemi microcomputer sono privi di pannello. Se si ha l'intenzione di usare il sistema più che altro in linguaggio ad alto livello, il pannello frontale è semplicemente inutile. Persino se l'utente intende programmare il sistema in linguaggio a livello assembly, la non disponibilità di un pannello frontale può non essere uno svantaggio importante. L'utente che non ha mai conosciuto un pannello frontale, non saprà mai cosa sta perdendo. E ciò che egli sta perdendo non è estremamente importante poiché tutte le strutture di debug possono essere ottenute su di un terminale.

I pannelli frontali venivano forniti sui primi microcomputers (Altair, Imsai) poiché ci si aspettava che fossero usati in un ambiente di laboratorio tradizionale.

Generalmente, essi non vengono più forniti. Su una macchina commerciale, essi sono semplicemente inutili. Se l'utente programmerà in BASIC, il pannello frontale non sarà mai usato.

Bisogna sottolineare che l'assenza di un pannello frontale non rende impossibile verificare il contenuto dei registri interni. Il compito diventa semplicemente più oneroso e viene fatto dalla console (CRT o telescrivente).

Infine, un sistema senza pannello frontale ha bisogno solo di un interruttore a tasto ON/OFF: esso appare più bello e più facile da usare, ed è perciò, più attraente. Per queste ragioni, la maggioranza dei microcomputers oggigiorno, non ha più un pannello frontale.

Quale bus?

La maggior parte dei microcomputer oggigiorno forniscono una connessione interna per mezzo di un "bus" standardizzato. Questo bus ha da 50 a 100 linee e trasporta i segnali necessari tra i boards del sistema. Il più popolare è il bus S100 intro-

dotto originariamente da Altair e poi usato da Imsai e molti altri. È basato sull'8080, e può essere usato con uno Z80. Non può essere usato da un 6800 in nessun modo che sia giusto. Esiste una certa quantità di altri bus come l'SS50 (per il 6800) l'IEEE-488, ed altri. I rispettivi meriti di questi bus sono valutati nel riferimento C207. Parlando genericamente, se sono necessarie semplicità e comodità nel collegare dispositivi addizionali, l'S100 ha un vantaggio preciso perché è di gran lunga quello più ampiamente disponibile. La maggioranza dei boards comuni sono ora disponibili con un'interfaccia per l'S100 talvolta per altri bus. Anche se la scelta può essere un po' più discriminata per bus diversi dall'S100 questa non è una restrizione significativa a causa dell'ampia scelta disponibile.

In breve, se si ritiene che la flessibilità nella scelta di una periferica sia un grosso vantaggio, l'S100 si può ritenere adatto. Tuttavia, generalmente un bus standardizzato non ha bisogno di avere un impatto importante sulla scelta di un sistema, se è disponibile un sistema completo da parte del costruttore.

I connettori sul retro

Sono desiderabili almeno tre connettori:

1 - Il teletype 20mA currente loop
Questo è un port classico seriale per un TT4, o altri terminali lenti. Vengono usati vari connettori a 4 fili.

2 - L'interfaccia standard RS-232C
Indispensabile. Questo connettore può essere usato con qualsiasi dispositivo equipaggiato con un'interfaccia EIA (RS-232C). Sarà quasi sempre usato per CRT. Allo stesso port possono essere collegati vari terminali in parallelo.

3 - Un'interfaccia per stampante parallelo.
Uscita a 8 bit paralleli specializzata per una stampante veloce. *Attenzione*: La maggior parte dei microcomputers hanno un unico port I/0 *seriale*. In questo caso sia il connettore RS-232C, e la presa per il circuito chiuso di corrente da 20mA sono connessi ad esso in parallelo.

Questo significa che due dispositivi, come un CRT ed una telescrivente, possono essere *collegati* ad una presa, ognuno simultaneamente, ma di solito non possono essere *usati* simultaneamente. Sull'ingresso, chiaramente, entrambe le tastiere non dovrebbero essere usate nello stesso tempo.

Sull'uscita essi possono restare connessi allo stesso tempo se entrambi funzionano alla stessa velocità (110 baud per una TTY). Comunque, un CRT è di solito predisposto per un baud rate più alto (fino a 9600 baud).

La disponibilità di entrambi i connettori RS-232C ed il 20mA è una *comodità*. Non significa che due terminali possano essere collegati e funzionino simultaneamente (questo richiederebbe due ports seriali).

D'altra parte, e a condizione che il microprocessore sia equipaggiato con un port parallelo, una stampante parallela ad alta velocità può funzionare simultaneamente.

Kit o assemblato

Un kit di solito offre notevole risparmio all'acquirente potenziale quando viene comparato a sistemi non-assemblati (con le varie parti non unite tra loro). Se l'intenzione è quella di risparmiare soldi allora si può acquistare un kit ed il valore istruttivo di assemblare il sistema, come pure la soddisfazione psicologica, valgono il tentativo. Tuttavia nel caso di una scatola microcomputer, bisogna sottolineare che la qualità dell'assemblaggio reale è importante per l'affidabilità del sistema che è stato costruito. Per questa ragione, un utente inesperto potrebbe cavarsela meglio comprando un sistema assemblato piuttosto che spendere lunghe notti nel tentativo di individuare connessioni o componenti difettosi. È probabile che non si realizzi il risparmio nella costruzione di un kit se l'utente non ha a disposizione un equipaggiamento per effettuare test adeguati. Strumenti di collaudo adeguati possono essere molto costosi e spazzare via ogni vantaggio finanziario ottenuto dal lavoro di assemblaggio. La mancanza di tali strumenti può portare ad errori nel funzionamento del sistema che si verificheranno solo dopo quando verranno incontrate combinazioni inusuali e rovineranno il valore del sistema o la sua affidabilità.

Recentemente è nato un altro problema: componenti non affidabili. Sono spariti gli scarti da parte dei principali costruttori di semiconduttori, sono stati risiglati, e sono apparsi sul mercato dei consumatori.

A questo proposito può essere fatto poco. Il problema di un kit è che, se esso non funziona, l'utente deve individuare i componenti difettosi. Con un sistema assemblato, la situazione è piuttosto migliore. Comunque, i componenti difettosi possono anche guastarsi solo dopo un anno, o persino a temperatura alta.

A favore dei microcomputers bisogna notare che:

1 - Essi sono intrinsecamente più affidabili dei minicoputers tradizionali, perché essi hanno molte parti in meno.

2 - Essi sono facili da fissare, e la spesa per rimpiazzare un intero board è piccola.
In sommario: state attenti ma non siate paurosi. È meglio che voi impariate a programmare piuttosto che a saldare. Comunque il risparmio di un kit potrebbe rendere accessibile a voi il microcomputer, laddove non fosse altrimenti.

SOMMARIO

Ora è stato introdotto un certo numero di criteri di scelta ed a sua volta ognuno è stato qualificato secondo la sua importanza. Cosa resta? Per lo più il supporto software.

Il supporto software è probabilmente la chiave nella scelta di un sistema. Se si intende usare un sistema in una vasta gamma di applicazioni, alla maggior parte degli utenti non interessa dover riprogrammare il sistema ogni volta. Essi vorrebbero la disponibilità immediata del software (dei packages che essi possano inserire nel sistema, ed usare per le loro applicazioni). La disponibilità di una vasta libreria di programmi, comodamente usabili su un sistema, è probabilmente una delle chiavi

principali nella scelta riuscita di un sistema. In aggiunta, il sistema dovrebbe offrire spontaneamente le strutture hardware necessarie, e può essere richiesta la facilità di aggiungere dispositivi nuovi per applicazioni specifiche volute dall'utente.

Cosa pensare del microprocessore XYZ?

L'ipotetica disponibilità di un nuovo microprocessore XYZ, due volte più veloce, con il doppio di registri, ecc., non ci deve riguardare qui. A meno che questo nuovo microprocessore possa usare i bus standard ed il software standard che già esistono, esso non avrà un impatto significativo sul mercato del personal computer prima di un lungo tempo. Un microprocessore da solo non fa un sistema. Si é visto che la scelta di successo sta nel conciliare simultaneamente un certo numero di criteri svariati. Perciò, la natura del microprocessore è di piccolo rilievo finché si soddisfano i criteri che sono già presentati.

Poiché si è stabilito che le periferiche hanno un grosso impatto sulle prestazioni del sistema, e sul suo costo, ora esaminiamole.

CAPITOLO 9
LE PERIFERICHE

La scelta di periferiche adatte può essere più complessa che non la scelta della "struttura principale", cioé del microcomputer vero e proprio. Esso ha pure di solito il maggiore impatto sul costo e sull'utilizzabilità del sistema.
In questo capitolo, saranno esaminate e valutate tutte le periferiche usuali per prima cosa i dispositivi di input (la tastiera), poi i dispositivi, di output (display, stampante), dispositivi di memoria di massa (disk, nastri), ed infine dispositivi speciali (input-output voce, light pen, ed altri).

LA TASTIERA

Spesso la tastiera è incorporata nella stessa struttura principale, o anche nell'unità di display CRT. Tuttavia, la maggior parte delle tastiere sono unità separate.
Esistono relativamente poche possibilità nella scelta di una tastiera e ciò è una fortuna. I criteri di scelta per una tastiera ricadono nell'area del senso comune. Deve essere resistente, affidabile, ed offrire buoni contatti. La disposizione dei tasti deve essere pratica per l'applicazione considerata. Una caratteristica da ricercare è la protezione *da rollover multiplo dei tasti.* Il problema accade quando vengono premuti simultaneamente diversi tasti, o quasi simultaneamente. Questo accadrà, per esempio, durante l'introduzione rapida di dati. Semplicemente la tastiera può ingoiare, rifiutare o scartare i tasti addizionali premuti. Una tastiera "protetta" immagazzinerà automaticamente le chiusure multiple dei tasti fino a che essi non saranno più simultanei.

Per l'utente commerciale: una semplice tastiera alfanumerica non è sufficiente una "tastiera commerciale" completa deve avere:
1 - usuali tasti alfanumerici
2 - una tastiera numerica separata con le 10 cifre, il controllo di cursore per lo schermo (su, giù. destra, sinistra), più tasti di controllo (reset, ecc.).

DISPOSITIVI DI USCITA

Il display CRT

Il display CRT è diventato il mezzo universale di uscita per i sistemi a microcomputer. Questo perché esso è silenzioso e rende visibili i risultati.

Il monitor video

Un monitor video è semplicemente un televisore senza i componenti elettronici di sintonizzazione, separazione ed amplificazione (assieme all'altoparlante, i selettori di canale, ecc.). Poiché esso non ha bisogno di rivelare frequenze UHF/VHF, un monitor ha una connessione di entrata diretta sul video, e può accogliere segnali di larghezza di banda molto maggiore che non nel caso di un regolare TV. In particolare, sono possibili 80 linee di caratteri.

Ogni TV comprende un monitor e può essere modificato per ottenere entrata diretta sul video. Comunque, questo è l'argomento di un altro libro (C207).

Il terminale video display

Il terminale VDT (Video Display Terminal) o CRT è un terminale costruito con un monitor, una tastiera, l'elettronica di controllo (per controllo del cursore, controllo della tastiera, funzioni dello schermo, e memoria locale). È stato il terminale tradizionale per comunicare con sistemi di computer, ed è uno dei più adatti in un

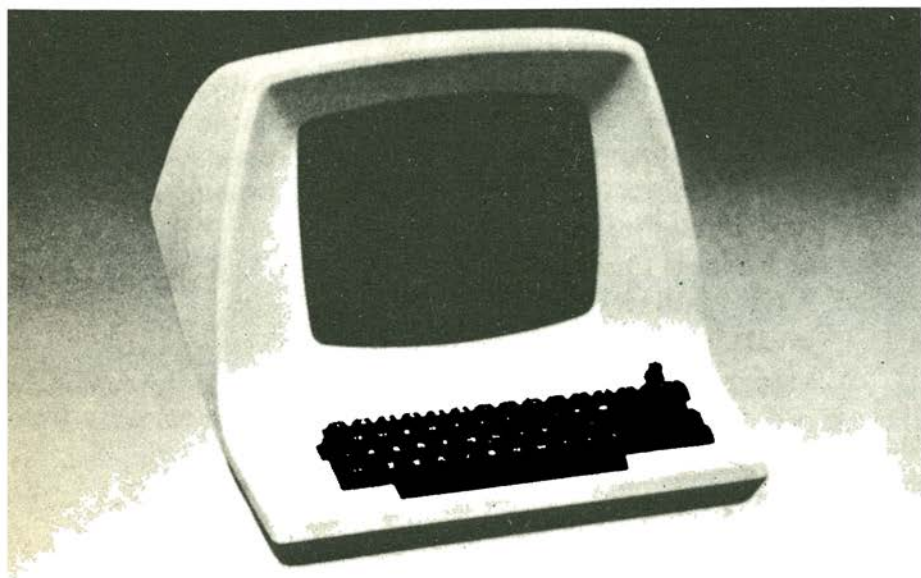



Fig. 9-1: Il "Terminale Non Intelligente" della Lear Siegler è un terminale CRT fondamentale. Esso presenta uno schermo da 12 pollici, 59 tasti, 24 righe di 80 lettere, connettori RS232 e 20 mA (64 caratteri ASCII, matrice di punti 5 × 7).

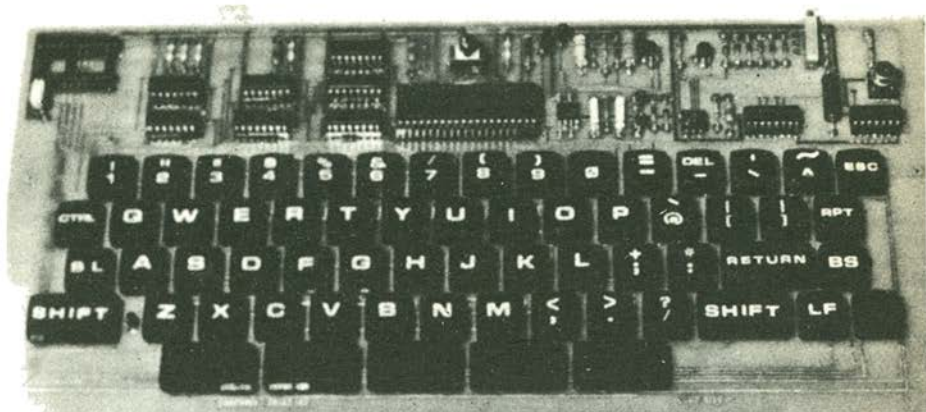



Fig. 9-2: Tipica tastiera QWERTY ed elettroniche di decodifica.

ambiente commerciale a causa della sua progettazione professionale e caratteristiche di comodità. Tuttavia, è l'alternativa più costosa (700 dollari ed altre, a seconda delle caratteristiche).

Una combinazione di tastiera più Monitor Video più software può fornire servizi equivalenti. Grande quantità di testo in un breve periodo di tempo, muovendo il testo verticalmente (questo è chiamato scrolling-arrotolare). Per di più un monitor CRT può essere acquistato per meno di 100 dollari così che può essere facilmente integrato in un progetto.

Si possono distinguere tre tipi di displays:

1 - Il televisore tradizionale
2 - Il semplice monitor
3 - Il terminale display video (VDT)

Ora esaminiamo queste tre possibilità.

Il televisore tradizionale

Nella misura in cui esso può essere disponibile "libero", questo può essere il più economico. Tuttavia, se uno si collega direttamente all'antenna (come nei giochi televisivi) le limitazioni di larghezza di banda derivanti darebbero come risultato linee di caratteri corte sullo schermo.

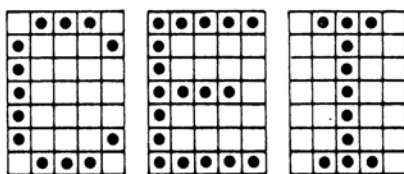
Comunque, nel caso di display a colori, le alternative sono talmente costose da rendere il televisore una scelta privilegiata.

Si deve ricordare che la limitazione delle linee corte resta. Considerando il costo molto basso di un monitor in bianco e nero (e la sua ampia larghezza di banda) non c'è in pratica nessun vantaggio nel prendere un televisore in bianco e nero.

Fig. 9-3: Il Soroc IQ 120 è un terminale CRT fondamentale che compete con quello precedente. Esso presenta 24 righe di 80 caratteri, pad numerico, maiuscole e minuscole.

Visualizzare un testo su di un display

Esaminiamo ora le caratteristiche comuni di tutti i displays. Lo scopo di ogni display è quello di effettuare il display di un testo, e se è possibile, permettere alcuni grafici.

Le immagini vengono visualizzate su uno schermo televisivo tradizionale illuminando piccoli punti sullo schermo per visualizzare i caratteri. Semplicemente i punti sono molto più larghi così da avere caratteri "visibili chiaramente". Ogni carattere è definito da una *matrice di punti.* Vediamo un esempio. Ogni carattere viene visualizzato illuminando i punti in un rettangolo che ha, per esempio, 7 righe di 5 punti. Per avere una definizione migliore vengono usati più punti, per esempio, matrici di 7 × 9 punti. Tuttavia questo riduce il numero massimo di caratteri di una certa dimensione che possono essere visualizzati sullo schermo. Su sistemi a basso costo, viene usata la matrice 5 × 7. Tuttavia, è difficile distinguere le maiuscole dalle minuscole ed estendere le lettere minuscole al disotto della riga con così pochi punti.

Fug. 9-4: Una matrice di punti 5 X 7

La seguente è una domanda tecnica interessante: si è mostrato che un CRT riceve caratteri dal computer sotto forma di bytes, cioé gruppi di 8 bit (ogni carattere corrisponde ad un'unica combinazione di 8 bit nel codice ASCII).

La domanda è: *come vengono convertiti questi 8 bit in 35 o più punti sullo schermo?*
Risposta: *semplicemente usando dei chip di memoria (ROM). Per ogni codice di 8 bit, un certo numero di locazioni di memoria forniranno la disposizione di punti.*
Conseguenza: poiché è facile rimpiazzare i chip ROM, è semplice modificare l'insieme dei caratteri. Display professionali possono offrire insiemi di caratteri sostituibili, estendendosi dall'APL a linguaggio esteri.

Quante righe e caratteri?

Chiaramente l'utente vorrebbe più caratteri, e righe possibili. Nel caso di un televisore dove il computer si collega all'antenna, la larghezza di banda limitata di un televisore limita l'utente a 24 linee di 40 caratteri, usando un televisore di buona qualità, una matrice 5 × 7, e solo caratteri maiuscoli. È comune un'ampiezza di 24 caratteri.

Nel caso di un monitor, o di un VDT, il computer si collega direttamente all'ingresso video, migliorando considerevolmente la larghezza di banda. Sono quasi standard 24 linee di 80 caratteri (maiuscoli e minuscoli).
Per l'utente commerciale: la lunghezza standard di una riga per applicazioni commerciali è di 80 caratteri (l'ampiezza di una scheda IBM) e idealmente dovrebbe essere di 120 e 132. Un televisore non è accettabile, a meno di modifiche per collegamento diretto al video, nel qual caso è usato come monitor. Per scopi commerciali, ma minima lunghezza deve pure essere di 24 righe.

Stampanti standard stampano 80 caratteri per linea o persino di più (132), di qui l'ovvio vantaggio di avere una riga di 80 caratteri.

Caratteristiche addizionali del CRT

La maggior parte delle caratteristiche addizionali sono realizzate da un programma, o talvolta da una scheda di controllo più complessa:

- il cursore: ogni CRT deve visualizzare un cursore per indicare una posizione sullo schermo, di solito la posizione dove apparirà il prossimo carattere. La forma

di un cursore é di solito un quadrato o una underline (linea di sottolineatura).La sua forma può essere programmabile, e può lampeggiare.

- il sistema può offrire due tinte, grigia e bianca, adatte a separare le domande dalle risposte, o anche il contrario (nero su bianco).
- i caratteri o il cursore possono essere fatti lampeggiare.
- possono essere disponibili grafici limitati. Di solito essi sono realizzati scegliendo punti specifici della matrice di punti usata per visualizzare i caratteri.
- il colore è un'ovvia opzione. Se si usa un televisore a colori, di solito sono disponibili quattro "colori", nero, bianco, viola, verde.

Stupido contro intelligente

Caratteristiche addizionali possono essere ottenute trasferendo al terminale display alcune delle funzioni eseguite dal microcomputer. Invece di un "terminale stupido", esso diventa un "terminale intelligente".

Un terminale intelligente di solito è progettato per essere utile "fuori linea" cioè scollegato dal computer.

Esso offre una memoria locale per l'immagazzinamento dei dati e funzioni di editing (per correggere errori). A causa del basso costo dei microcomputer, i terminali CRT intelligenti tendono a restare inutilizzati in questo nuovo ambiente.

Sommario sul display

1 - *Per gli hobbisti:* per il colore l'unica opzione a basso costo è il televisore casalingo. Per il bianco e nero, un televisòre commerciale vi ridurrà ad avere 16 righe di 32 caratteri (approssimativamente), un *monitor* è raccomandabile.
2 - *Per l'uomo d'affari:* quando la scatola del microcomputer non include una tastiera, è necessario un *terminale.* Un terminale stupido è di solito sufficiente. Otterrete 24 righe di 80 caratteri, maiuscoli e minuscoli. Cercate un terminale con una tastiera completa. Uno schermo di 12 pollici viene considerato ottimale. Per una facile lettura è richiesta una matrice di punti 7 × 9 o migliore.

Se il mobile del microcomputer include la tastiera richiesta, voi potete semplicemente aggiungere un monitor da 12 pollici, più il necessario board di controllo per CRT (se disponibile) all'interno del microcomputer ed ottenere funzioni equivalenti.

Parlare al display

Sono stati inventati diversi dispositivi di input sulla tastiera per indicare locazioni specifiche. Quelli più frequentemente usati saranno descritti alla fine di questo capitolo. Essi sono: light pen, joystick, e in misura minore, il "mouse" (topo).

La stampante

Un display CRT fornisce uno "soft copy", ma può muovere le informazioni attraverso lo schermo ("scrolling"), o persino "flip pages" ("paging" togliere le "pa-

gine" dalla memoria centrale per metterle altrove per liberare parte della memoria per altre "pagine" di dati). È di solito desiderabile, o indispensabile, in un ambiente commerciale, fornire uno "hard copy" (copia materiale) su carta. Una stampante è necessaria in quasi tutti i casi.

Perché non usare proprio la Selectric dell'ufficio? Modificata in maniera adatta, può essere usata. Tuttavia è costosa, relativamente lenta (30 caratteri per secondo, o "cps") e può non essere sufficientemente affidabile, a meno che sia progettata come terminale di un computer. Questa opzione sarà riesaminata sotto.

Perché non fotografate il display CRT? Questo non sarebbe molto pratico, a causa delle fotografie molto piccole. Tuttavia è tecnicamente fattibile l'accoppiare un display ad una copiatrice per ufficio, e questa potrebbe diventare una alternativa in futuro.

Consideriamo ora la gamma delle alternative. Naturalmente, noi vogliamo:

1 - basso costo
2 - alta qualità di stampa
3 - velocità
4 - affidabilità
5 - basso rumore

La scelta finale, come al solito, sarà un compromesso. Si possono distinguere quattro tipi principali di stampanti:

1 - Termica/Elettrosensibile
2 - Stampante a nastro
3 - Stampante a matrici
4 - Line Printer.

Esse saranno ora esaminate in funzione degli obiettivi appena stabiliti.

Stampante termica ed elettrosensibile

Le stampanti termiche usano una carta speciale ed imprimono a caldo i caratteri su essa. L'elemento stampante ha punti o segmenti che premono sulla carta e viene fornita energia per incidere il carattere richiesto all'appropriata combinazione di segmenti.

Siccome l'elemento stampante è semplice e non ha parti in movimento, queste stampanti sono economiche e silenziose. Tuttavia esse offrono una qualità di stampa mediocre (comparata ad una buona Selectric), non possono funzionare ad alta velocità (a causa del tempo richiesto per incidere i caratteri), e richiedono carta speciale.

Stampanti elettrosensibili e termiche sono di solito usate come stampanti dal costo più basso rivolte ad una capacità minima (piccola larghezza della riga, bassa velocità). Tuttavia, ci sono eccezioni degne di nota come il Texas Instrument Silent Terminal, ed esse possono essere usate fino a 90 cps.

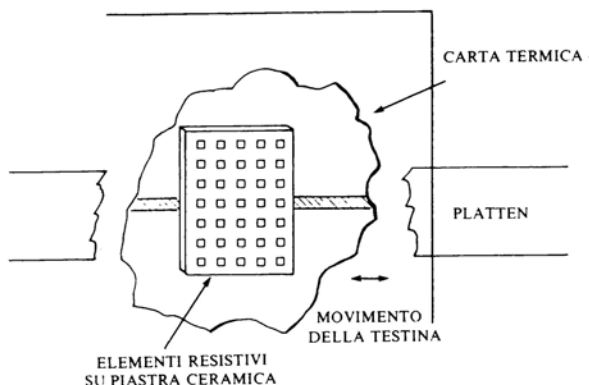



Fig. 9-5: Stampante termica con matrice di punti.

Stampante a nastro

Le stampanti ad *impatto* sono quelle usate più frequentemente. Il loro principio di funzionamento è di usare una testina stampante e di smuoverla, o anche testine multiple.

Una stampante a nastro funziona essenzialmente come una Selectric per ufficio, muovendo la testina attraverso il foglio. Sono usate tre tecniche principali per la testina:

1 - un elemento sferico o cilindrico con tutti i caratteri
2 - una daisy wheel (ruota a margherita)
3 - una matrice di aghi. Quest'ultimo argomento sarà studiato nella prossima sezione.

Esaminiamo ora i primi due elementi di stampa.

La testina cilindrica

Questa è la tecnica usata da uno dei più vecchi terminali per computer, la Teletype (telescrivente - un'abbreviazione di "teletypewriter" - nome registrato). Viene usata una testina stampante cilindrica (vedi fig. 9-6). Il cilindro ha 4 righe di caratteri, può ruotare in 17 posizioni, e può essere alzato. Per stampare un carattere, il cilindro viene ruotato nella posizione appropriata, sollevato al livello della riga necessaria, e quindi colpito con un martello. Il carattere preme il nastro contro la carta. Tutti i collegamenti sono elettromeccanici ed usano calamite realizzate con elettromagneti.

La Teletype KSR33 standard funziona a 10 cps. È rumorosa, lenta ma affidabile, di basso costo (un po' più di 1000 dollari) ed ha una leggibilità eccellente. È anche

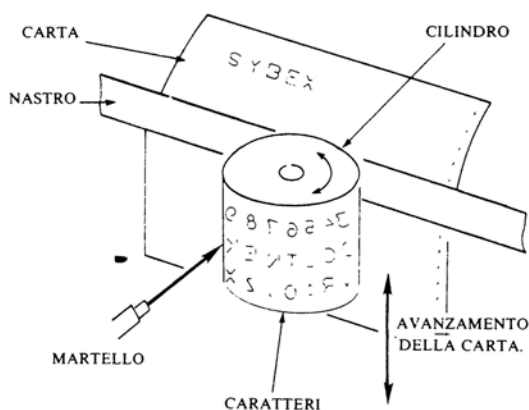



Fig. 9-6: Cilindro stampante della telescrivente.

limitata alle maiuscole per il suo prezzo, è ancora una buona scelta come stampante lenta, leggibile ed affidabile. Non può essere presa in considerazione per applicazioni commerciali.

Esistono molti altri modelli. In particolare, la ASR33 (ASR sta per "Automatic Send Receive" - trasmissione ricezione automatiche - mentre KRS sta per "Keyboard Send Receive" - trasmissione ricezione con la tastiera) é una KSR33 equipaggiata per un basso costo aggiuntivo con un perforatore- lettore di nastro di carta. Questo era di valore come "hard copy" dei programmi al tempo in cui non esistevano i floppy disk, e può ancora essere usato su piccoli sistemi.

Le due interfacce standard per la Teletype sono l'RS-232C ed il 20mA current loop. Essa usa il codice ASCII.

La Selectric

La Selectric è una macchina da scrivere convertita dalla IBM. Essa usa una sfera di plastica metalizzata che può colpire e ruotare. È ancora lenta (da 15 cps a 30 cps), ma offre, una qualità di stampa identica a qualsiasi macchina da scrivere da ufficio. Il suo costo è un po' più alto di quello della Teletype, ma può essere usata in un'applicazione commerciale minima. Ha maiuscole e minuscole e elementi stampanti sostituibili. È comunque inadatta per funzionamento continuo, o liste lunghe. Continuiamo la nostra ricerca.

La stampante a daisy wheel

La stampante a daisy wheel (ruota a margherita) usa una ruota con fino a 100 caratteri (vedi fig. 9-8). Ogni carattere è su un proprio braccio individuale, così che è

Fig. 9-7: La telescrivente tradizionale

necessaria solo una rotazione. Il martello colpisce solo una lettera singola. In aggiunta, è possibile garantire un posizionamento accurato ed una battitura uniforme.

Questo permette un funzionamento veloce, silenzioso ed affidabile, con una superba qualità di stampa.

In fig. 9-8 appare l'immagine di una vera daisy wheel. In aggiunta le ruote sono

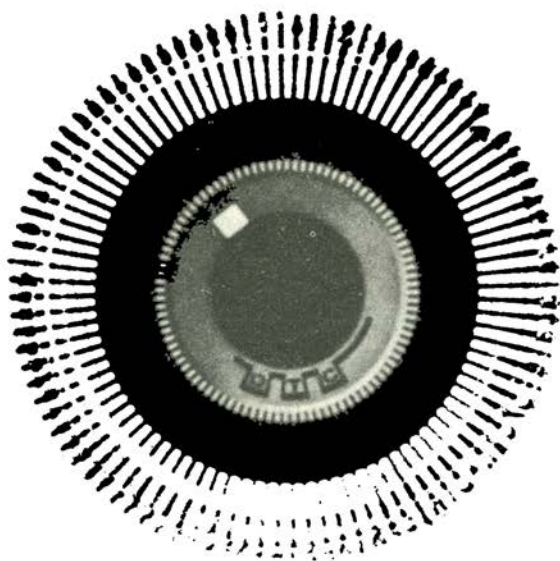



Fig. 9-8: La daisy wheel mostra i singoli caratteri

sostituibili, così che possono essere usati vari insiemi di caratteri, e sono standard per diversi costruttori (Diablo, Qume, Hewlett-Packard).

La qualità di stampa è così alta che viene spesso fornito l'allineamento del testo dando come risultato una copia pronta ad essere fotografata.

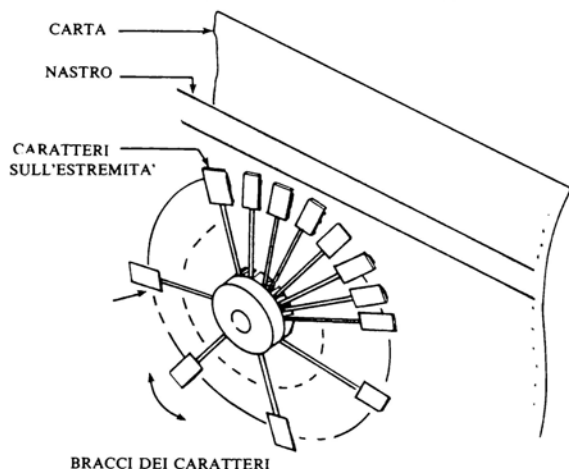



Fig. 9-9: La daisy wheel in funzione

Per l'utente commerciale: la stampante daisy wheel è l'attuale stampante ottimale per un sistema a microcomputer, ma costa da 3000 a 5000 dollari, probabilmente il componente più costoso del sistema.

Stampate a matrici

Una stampante a matrici usa aghi per stampare una configurazione di punti sulla carta. I caratteri sono generati proprio come nel caso del display CRT, scegliendo l'appropriata combinazione di aghi in un rettangolo di 5 righe con 7 colonne o 7 righe con 9 colonne. Ogni ago è equipaggiato con una bobina (solevoide). Quando viene fornita energia l'ago è lanciato contro il nastro, stampando un punto sulla carta. Per ogni carattere dovranno essere stampate sette o nove linee di punti. Tuttavia, l'inerzia è minima, così che è possibile un funzionamento veloce.

In un sistema a basso costo, viene usata una tastiera singola, che stampa carattere per carattere, e si sposta attraverso il foglio.

In sistemi dal costo più alto, è disponibile un'intera linea di aghi per 80 o 120 caratteri. Un'intera linea verrà stampata in appena sette o nove fasi. Questo é illustrato in fig. 9-13.

I vantaggi generali sono: costo relativamente basso, alta velocità, funzionamento silenzioso. Lo svantaggio principale è la qualità relativamente scadente della stampa. Bisogna sottolineare che la qualità di stampa è sufficiente per l'occhio, ma non per la riproduzione di lettere commerciali.

Una stampante a matrici famosa è la Decwriter, che offre il funzionamento a 30 cps, e di solito alta qualità.

Per l'utente commerciale: la stampante a matrici è una buona alternativa alla stampante daisy wheel a quasi la metà del prezzo, purché non si abbia bisogno di generare lettere commerciali.

Fig. 9-10: La Diablo Hyterm é una stampante a daisy wheel con 94 caratteri 45 cps.

La stampante a catena

Un tipo di stampante che non é stato mai nominato qui è la stampante a catena. Le stampanti a catena possono ottenere una velocità di più di 1000 righe al minuto. Esse sono usate per grossi computer, e talvolta per minicomputer. Tuttavia, il loro costo è semplicemente così alto, che non avrebbe senso collegarle ad un microcomputer di basso costo. Se l'utente investe nel costo molto alto di una stampante a catena ad alta velocità, dovrebbe pure orientarsi verso un grosso potente computer per guidarla.

La stampante a catena mostrata in fig. 9-15 illustra un tipo a tamburo. Il carattere viene spostato attorno al tamburo fino alla posizione di stampa richiesta.

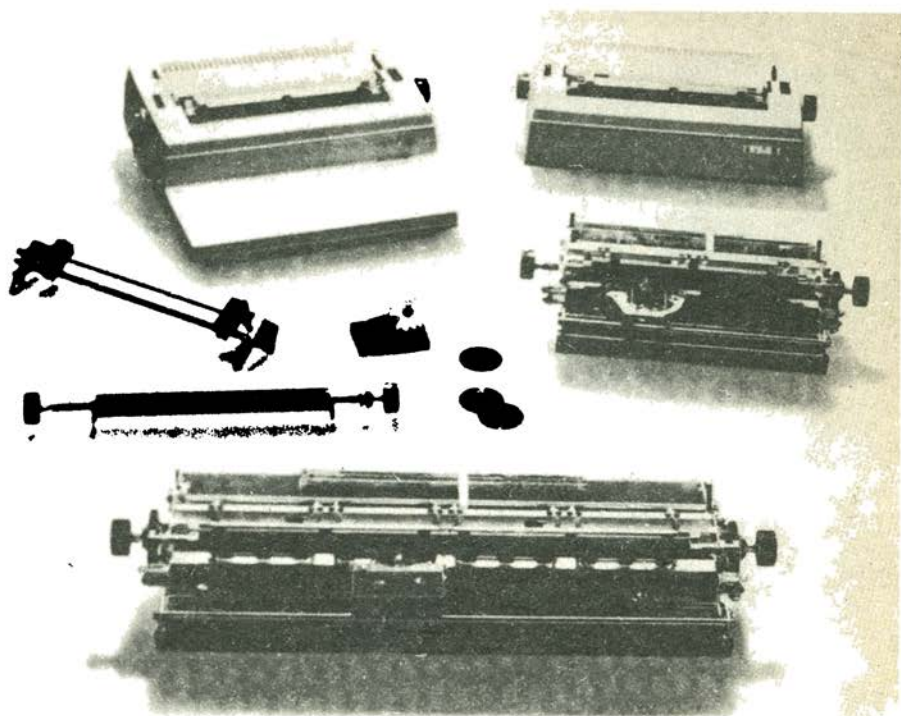



Fig. 9-11: La famiglia Qume Sprint Micro 3 offre tipi di dajsy wheel a basso costo.

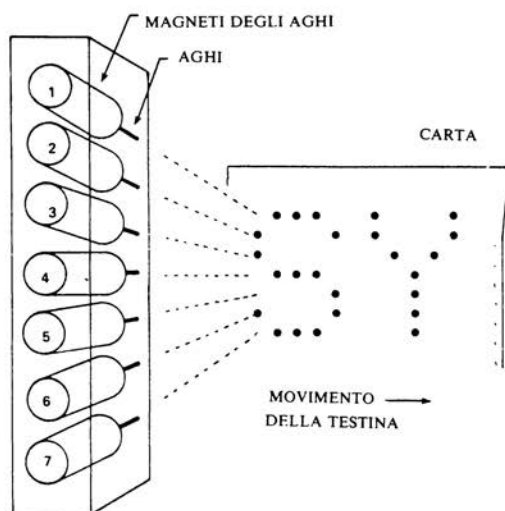



Fig. 9-12: Sette aghi possono colpire simultaneamente il foglio.

La stampante in linea

In contrasto alle 200 - 300 linee per minuto come massimo per le stampanti precedenti, una line printer (stampante di linea) funzionerà fino a 2000 Ipm (linee per minuto). Non sarà descritta qui perché il suo prezzo è semplicemente sproporzionato per un sistema a microcalcolatore.

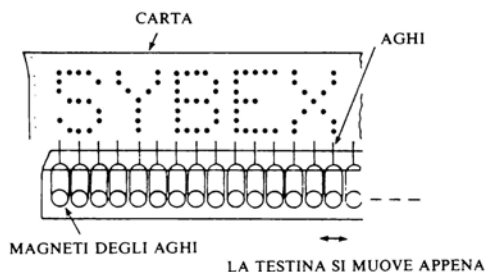



Fig. 9-13: Viene stampata immediatamente un'intera riga di punti.

Superstampanti

Nelle installazioni dei grossi computers, dove il prezzo non è più una considerazione, stampanti a non-impatto vengono usate per super-velocità. Una stampante a getto d'inchiostro funziona proiettando piccole gocce d'inchiostro e deviandole elettrostaticamente. Possono essere ottenute velocità di 40.000 linee per minuto.
Nella *stampante a laser*, la carta viene caricata elettrostaticamente, ed attira polvere d'inchiostro secca, come nella macchina Xerox (fotocopie). Il pattern viene quindi indurito sulla carta. Vengono stampate molte linee simultaneamente, e vengono raggiunte velocità di 20.000 linee al minuto.

Stampanti commerciali: un sommario

Sono state indicate le alternative essenziali: la migliore è la stampante daisy wheel, ma diventa la parte più costosa di un sistema. Le altre due alternative sono una Selectric, ed una buona stampante a matrici.
Un'"opzione" essenziale: è necessario un avanzamento dentellato della carta se deve essere stampato qualche modulo, etichetta, o assegno.Nessun meccanismo a frizione può fornire l'accuratezza verticale necessaria a posizionare i moduli correttamente.

I DISKS

Poiché la memoria principale, all'interno del contenitore del microcomputer, è limitata a circa 48K byte, ed è volatile, grandi programmi e files devono essere immagazzinati su di un mezzo permanente. Sono usati sia i disks che i nastri. Saranno

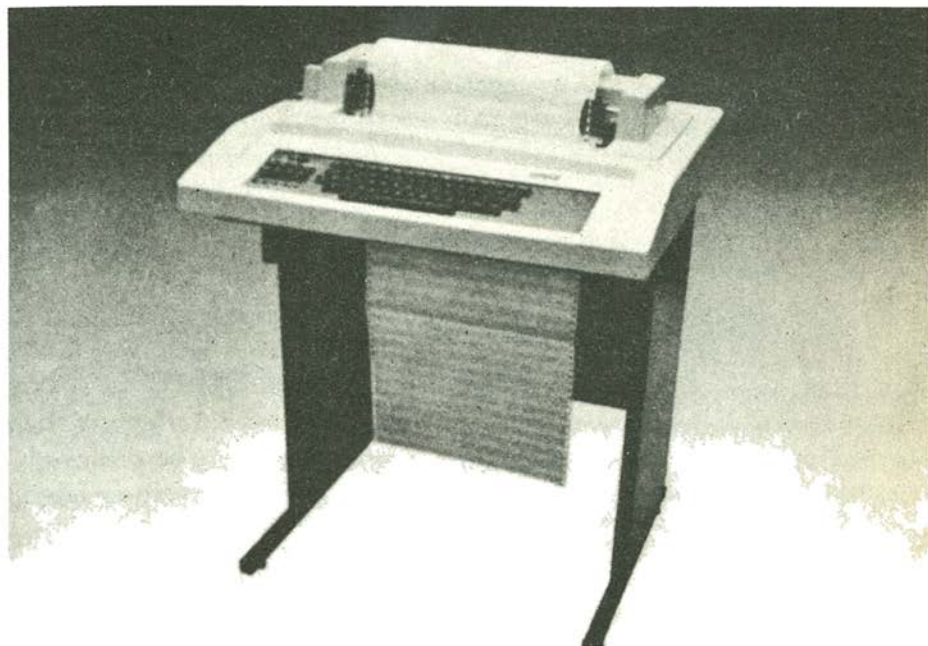



Fig. 9-14: La Decwriter LA-36 é una stampante a matrici di 30 cps di alta qualità. Essa usa una matrice di punti 7 X 7 ed ha un set completo di 128 caratteri.

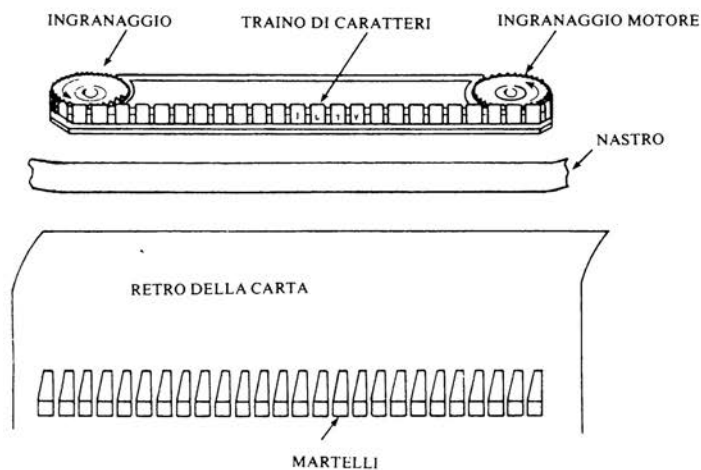



Fig. 9-15: Una stampante del tipo a tamburo.

presentati i rispettivi vantaggi di ognuno. Sarà mostrato che il "mini-floppy" è diventato la soluzione preferita per gli hobbisti, ma non è adeguato per la maggior parte delle applicazioni commerciali.

Il disk tradizionale per grossi computer é stato l'"hard disk" (disco duro): é un disk, rivestito su entrambi i lati con un ossido magnetico che ruota permanentemente. Una testina lettura/scrittura, molto simile alla testina di un registratore, viene posizionata sopra una "traccia" del disk. I dati possono essere "scritti" sul disk come sequenza di bit. Gli "0" e "1" sono scritti sulla superficie della traccia magnetizzando le particelle in una direzione o in un'altra. L'informazione immagazzinata in questo modo é "permanente", fintanto che non viene applicato un forte campo magnetico al disk.

Le tracce concentriche sul disk appaiono in fig. 9-17

Per poter recuperare le informazioni del disk, è necessario sapere su quale traccia sono immagazzinate, ed anche dove sulla traccia.

Per recuperare comodamente informazioni entro una traccia, è diventato uso standard dividere ogni traccia in settori. Questo è illustrato in fig. 9-18. Un settore tipico ha 128 byte. Le informazioni verranno ora recuperate per mezzo del loro numero di settore e numero di traccia.

Naturalmente, nessuno vuole preoccuparsi della reale assegnazione di settori ad un file mentre usa il sistema: un buon programma DOS (Disk Operating System) manipolerà tutte le utilizzazioni dei disk, e fornirà un FMS (File Management System - Sistema di gestione dei file) così che diventi possibile dire

"LOAD NEW FROM DISK"

ed i sei settori usati nel programma "NEW" vengono recuperati automaticamente dal disk. Questo viene chiamato "nominare simbolicamente un file".

L'hard disk sarebbe uno dei migliori mezzi di immagazzinamento di massa: esso offre alta velocità ad alta capienza. Sfortunatamente, è anche molto costoso, e non è stato ancora usato coi microcomputer. Tuttavia, degli "hard-disk" nuovi, più piccoli stanno diventando disponibili, e, quando saranno pronti, potrebbero essere la scelta preferita per l'utente commerciale.

Il Floppy Disk

Il floppy disk funziona proprio come l'hard disk, eccetto che è più piccolo e "floppy": il disk è morbido e contenuto permanentemente in una guaina di cartone. Il disco stesso ruota dentro la guaina. L'interno della guaina è rivestito di un materiale speciale a bassa frizione, ed ha un'apertura attraverso la quale la testina di lettura-scrittura può fare contatto col disk. Per maggiori dettagli sul funzionamento dei disks, il lettore interessato viene rimandato ancora ai riferimenti C201 o C207.

I formati dei floppy disks, al contrario degli hard disk, sono stati standardizzati (dalla IBM) ed i prezzi tendono ad essere simili per la maggior parte dei costruttori.

Il disk da 8 pollici originale fu introdotto dalla IBM con il 3740 alla fine degli anni 60.

Un floppy è un dispositivo meccanico complesso, e richiede un "board di controllo" che decodificherà i comandi provenienti dal microcomputers, e li eseguirà, come pure manipolerà il trasferimento di dati. Normalmente esso deve essere aggiunto alla scatola del microcomputer.

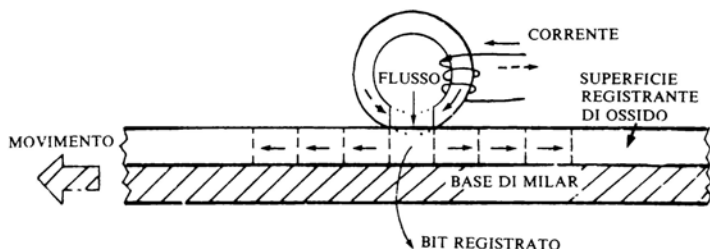



Fig. 9-16: Registrazione di un bit su un Disk.

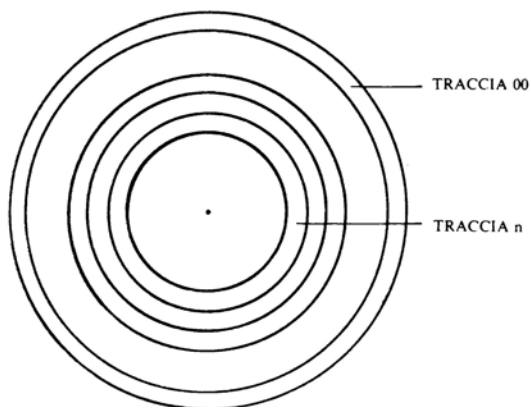



Fig. 9-17: Le tracce sul Disk.

Il Minifloppy

Il minifloppy è una versione più piccola del floppy: 5,25 pollici invece di 8. Fu introdotto dalla Shugart nel 76, ed essi stabilirono il formato dei dati (non IBM questa volta). Il minifloppy incontrò grosso successo nell'industria del microcom-

puter perché offre immagazzinamento di massa a basso costo (un diskette costa di per sé 5-10 dollari). Il floppy tradizionale ha da 35 a 40 tracce, a seconda del meccanismo di posizionamento della testina. Un costruttore (Micropolis) fornisce 77 tracce, cioé la doppia quantità di immagazzinamento, ed ancora sostiene di offrire un'affidabilità sufficiente.

Il "minifloppy normale" fornisce circa un terzo dell'immagazzinamento di un floppy regolare. Può essere usato con grande vantaggio in un ambiente hobbistico, e per mantenere programmi personali.

Per l'utente commerciale: la capienza di un minifloppy è troppo piccola. Il minimo richiesto è un floppy regolare. Persino un floppy regolare è scomodo per un grosso inventario o per una grande mailing list.

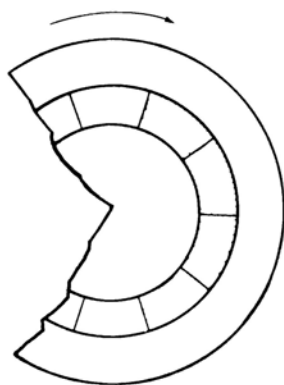



Fig. 9-18: Un file sequenziale di 4 Block viene immagazzinato in quattro settori.

Aumentare l'immagazzinamento

Sono disponibili due opzioni per raddoppiare l'immagazzinamento che si ha a disposizione: doppia intensità e doppia-faccia. Quelli a *doppia densità* raccolgono il doppio dei dati, ma perdono un po' in affidabilità.

Il floppy doppia-faccia usano entrambi i lati del diskette, ma introducono problemi meccanici addizionali.

Comunque, entrambe queste tecniche stanno maturando rapidamente e possono già essere usate con un'affidabilità accettabile.

Uno o due disk?

Se voi intendete copiare un programma o un file da un disk a un altro, ve ne servono due. Se avete bisogno di ordinare in ordine alfabetico o di mescolare due dis-

Fig. 9-19: Diskette e Minidiskette.

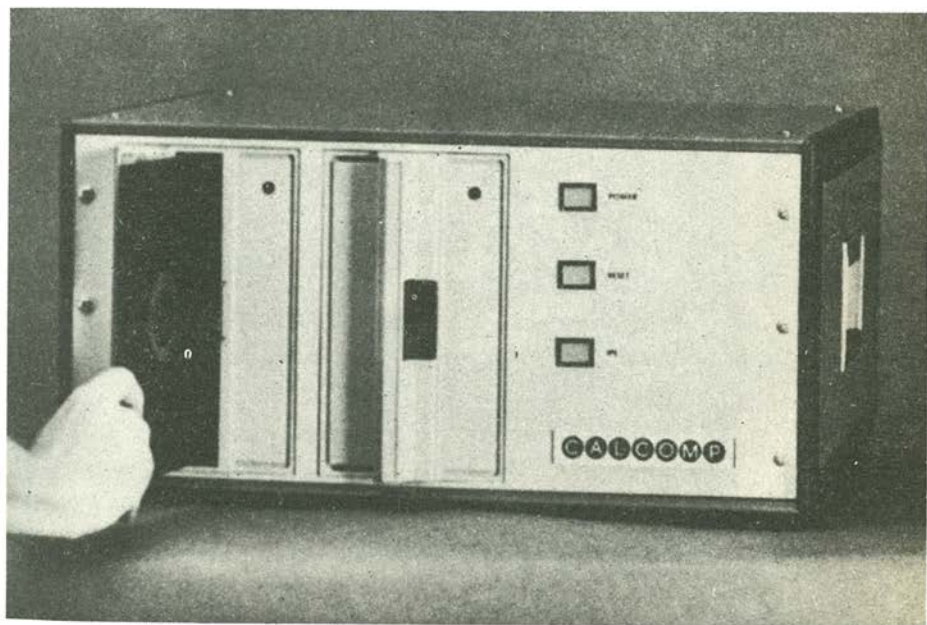



Fig. 9-20: Inserzione di un Floppy .

kette, ve ne servono due. Se avete bisogno di una maggiore quantità di immagazzinamento prontamente disponibile, vi servono due, tre o quattro disk.

Ogni sistema dovrebbe avere due disk. Qualsiasi cosa in meno limita le capacità del sistema, ma può essere accettabile per scopi di personal computing.

Per l'utente commerciale: Sono indispensabili due disk *come minimo.*

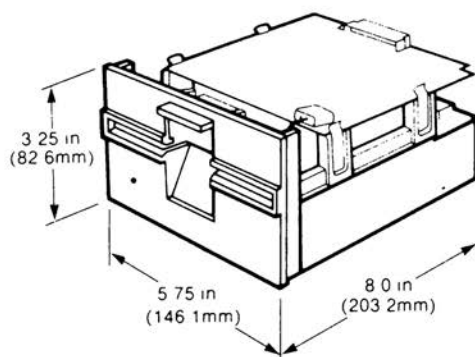



Fig. 9-21: Dimensioni del Mini-Floppy

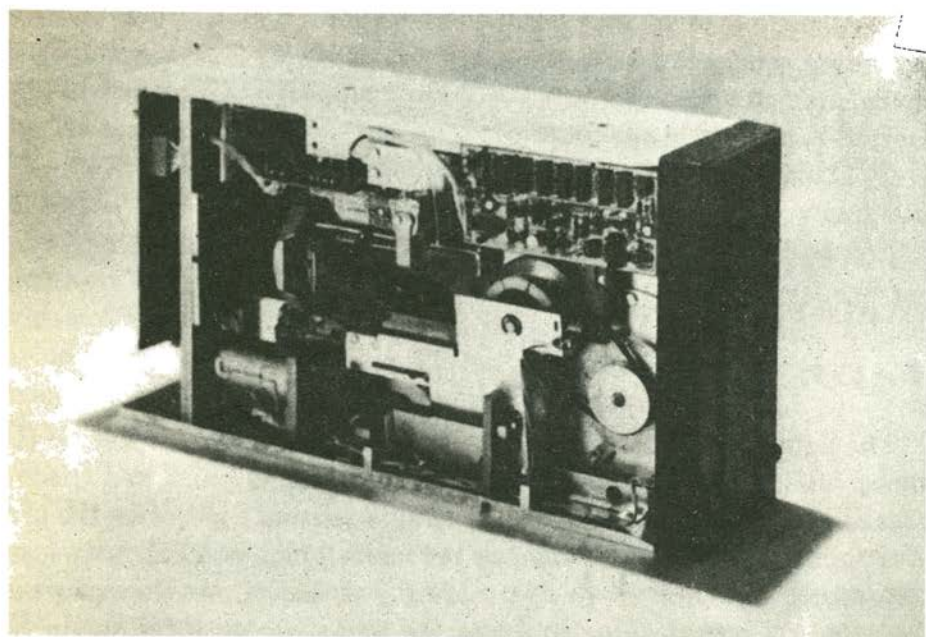



Fig. 9-22: Il Dual Diskette Drive 277 della Persci manipola due Diskette avendo le dimensioni di soli 8,6 × 4,4 × 15 pollici.

Sommario dei Disk

Il floppy (o mini-floppy) è diventato il dispositivo d'immagazzinamento di massa standard per i microcomputer. Un dual drive (guidatore di floppy doppio) è necessario per un funzionamento efficiente. Un mini floppy è adeguato per personal computing. Un floppy è il minimo necessario per un uso commerciale del computer, e può essere rimpiazzato nel futuro dal "mini hard disk" introdotto da poco.

NASTRI

Un registratore a cassette domestico può essere usato come mezzo di immagazzinamento di massa a basso costo. *Attenzione*: un registratore a cassette standard ha bisogno di un collegamento per ON/OFF a distanza, e deve essere disinserito durante il riavvolgimento e l'avanzamento veloce. Infine, il registratore a cassetta domestico costringe il segnale ad essere a frequenza sonora per avere una bassa distorsione armonica (ed ignora la distorsione di fase) il che porta a problemi di affidabilità, se viene usato un elevato ritmo di codificazione di bit per inch (per pollice di nastro).

Esistono anche registratori a cassetta digitali. Essi hanno ancora lo svantaggio di un lungo tempo di accesso, e la loro natura sequenziale (difficile accesso random-casuale).

MEMORIA DI MASSA FUTURA

Sono state sviluppate due tecnologie di memoria, e puntano a fornire una grande quantità di memorie (si dice un megabyte = 1 milioni di bytes) ad alta velocità e basso costo. Esse sono: CCD (Charge-Coupled Devices) e memorie a bolle. I CCD potrebbero sostituire fra qualche anno i mini-floppy. Le memorie a bolle possono offrire una memorizzazione più duratura in un prossimo futuro, ma ciò è ancora costoso.

ALTRE PERIFERICHE

La Light Pen

La light-pen (penna-luce) è un dispositivo di input da usare con un display CRT. È un dispositivo di comunicazione potente e comodo per indicare una locazione specifica sullo schermo. Funziona semplicemente percependo la luce quando l'area verso la quale essa è puntata viene illuminata dal raggio di luce che scandisce lo schermo continuamente. Nel momento in cui la luce viene percepita il computer (o piuttosto un programma) può calcolare approssimativamente la posizione sullo schermo.

La light-pen è particolarmente comoda per le selezioni: è sufficiente puntarla sulla scelta indicata.

Tuttavia, non offre buona precisione, ed è costosa, così che viene raramente usata con i microcomputers.

In un ambiente commerciale: una light-pen può non essere comoda; l'operatore comunica efficientemente col sistema per mezzo della tastiera e può *confermare* ogni scelta effettuata.

Se l'operatore dovesse puntare accidentalmente alla scelta sbagliata sullo schermo, sarebbe necessario ricominciare tutta la procedura di selezione per correggere ciò. Con una tastiera il programma aspetta il "Carriage Return" prima di eseguire, concedendo il tempo per la verifica, o correzione.

In un ambiente didattico: la light-pen è ideale, poiché un utente inesperto può essere dispensato dall'usare una tastiera, e gli errori non sono talmente dannosi. Un bambino, per esempio, può quindi puntare alla risposta "corretta" senza sapere nulla a proposito delle tastiere.

Il Joystick

Il joystick (leva di comando) è una leva verticale che può essere spostata a sinistra, destra, avanti, indietro o in qualsiasi posizione intermedia. È ideale per muovere un punto attraverso lo schermo in modo rapido. È usata estesamente per i giochi televisivi, ed è molto economica (e precisa).

Il Mouse

Il mouse (topo) è la versione più precisa della joystick. Essenzialmente è un piccolo dispositivo equipaggiato con ruote che può essere mosso con la mano in tutte le direzioni attraverso una tavoletta. Il movimento delle ruote può essere misurato con precisione, e un punto sarà mosso sullo schermo, nel momento in cui il mouse viene mosso. Il suo nome viene da un soprannome familiare dato a questo dispositivo negli anni 60 a causa delle sue dimensioni e dal tipo di movimento.

È costoso e di solito non usato con i microcomputer.

Tavoletta (RAND)

Esistono molti tipi di tavolette digitalizzate, dove uno può essenzialmente scrivere, o muovere una "penna" speciale. La posizione della penna viene percepita, e correlata ad una posizione sullo schermo. La buona risoluzione comporta un prezzo alto, così che le tavolette vengono raramente impiegate coi microcomputer.

Ingresso vocale

Si, è possibile dare comandi verbali ad un computer, con un adatto analizzatore della voce ed il suo programma. Entro la gamma di prezzi compatibili, coi micro-

computers, un tale sistema accetta un vocabolario limitato (diciamo poche dozzine di comandi) di comandi brevi ben pronunziati, dopo un "periodo di apprendistato", eseguito dal programma per ogni nuovo utente. Il tempo di processing necessario può dare come risultato che il comando sia eseguito solo pochi secondi dopo che il comando viene pronunciato.

Tali boards sono disponibili commercialmente, e possono essere aggiunti ad un sistema compatibile (generalmente S-100).

Uscita vocale

La voce può essere sintetizzata ed un programma può produrre una voce comprensibile in risposta a una codifica specifica del programma di analisi dei fonemi (unità di discorso elementari). Esso richiede un board sintetizzatore, un buon pro-

Fig. 9-23: Lo "Speechlab" della Heuristic offre l'ingresso voce. E' compatibile con l'S-100.

gramma di analisi che codifica i fenomeni quando pronunciati in un microfono, ed un vasto programma per la ripetizione. Il suono della voce può essere davvero buono. Il programma principale sta in una buona analisi della voce, nel codificare correttamente tutte le sue caratteristiche, così che può essere necessario un "ritocco" manuale.

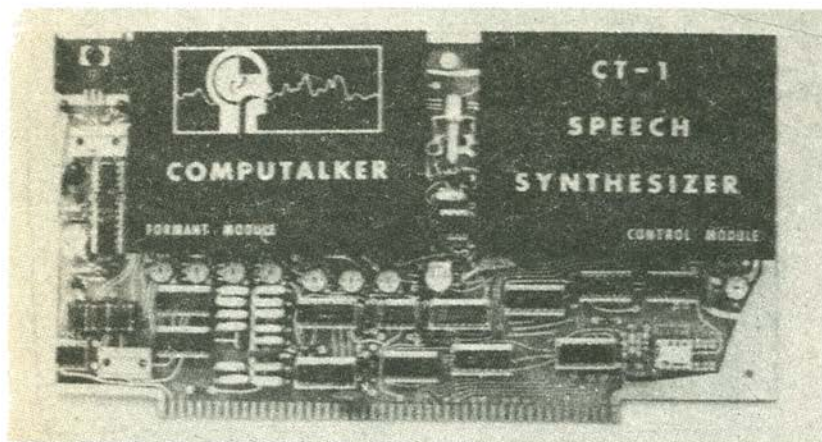



Fig. 9-24: Il computalker fornisce l'uscita vocale.

ALTRI DISPOSITIVI PER HOBBISTI

I LED

I diodi emittenti di luce sono il mezzo di display più economico, ma sono limitati a cifre numeriche o simboli esadecimali (da 0 a 9, da A ad F) se devono rimanere "economici". Essi sono usati su microcomputers ad "un board".

Interruttori

Interruttori sensitivi possono essere collegati direttamente al board di un microcomputer e sono usati per realizzare allarmi casalinghi contro gli scassinatori, o automazioni dei trenini elettrici.

Relé

I relé elettromeccanici forniscono un comodo isolamento tra due circuiti. Relé in miniatura possono ora essere montati direttamente sul board del microcomputer, permettendovi di chiudere un circuito che porta un voltaggio considerevole. Essi saranno necessari per accendere o spegnere luci, motori o altri dispositivi.

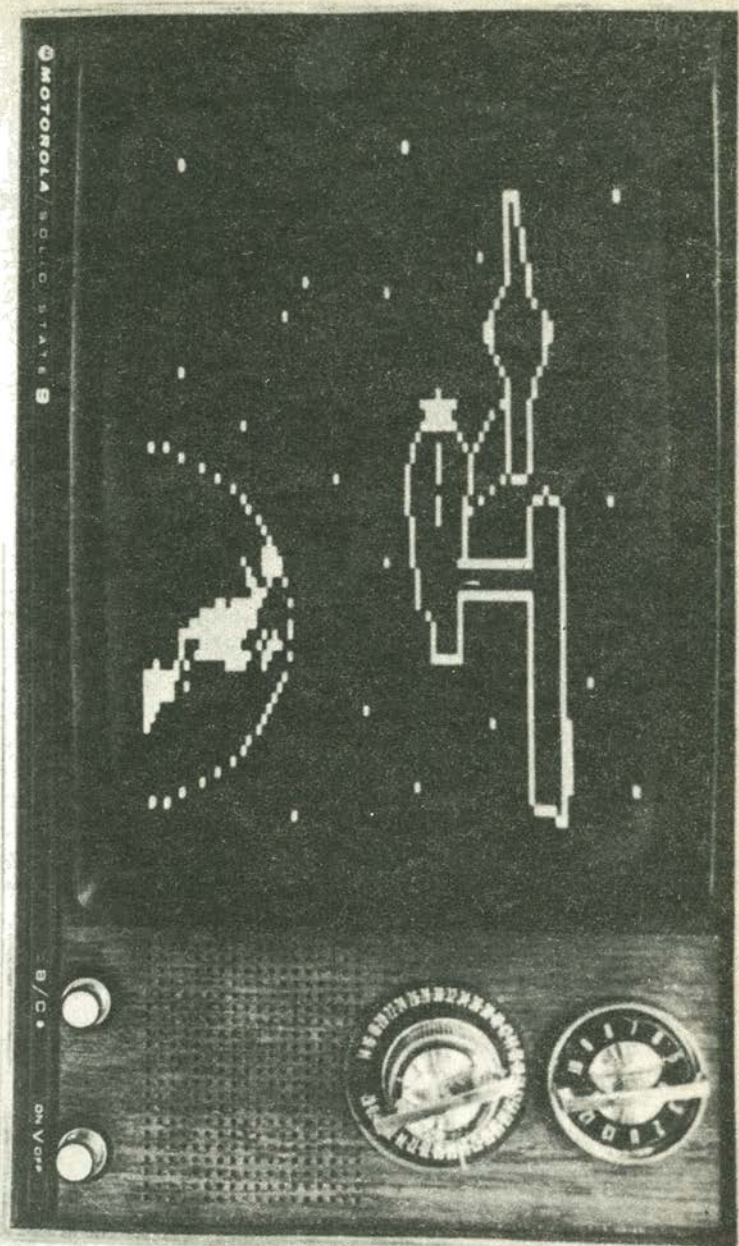



Fig. 9-25: U.S.S. ENTREPRISE nello spazio. Un televisore offre solo una capacità grafica limitata a causa della larghezza di banda limitata.

I DAC e gli ADC

Misurando una tensione con un voltmetro digitale per esempio si vedrà sul display: 12,5V. La tensione è una quantità analogica (che varia senza soluzione di continuità). 12,5V è un valore digitale che dà una misurazione approssimata con la precisione di 0,1V. Un DAC è un Digital to Analog Converter (Convertitore da Digitale ad Analogico). Sarà necessario se voi volete che il vostro computer produca un segnale analogico che può variare (quasi con continuità).

Un ADC è un Analog to Digital Converter (Convertitore da Analogico a Digitale), ed è stato necessario per il vostro voltmetro. Un ADC sarà necessario per la misurazione di qualsiasi quantità fisica: temperatura, pressione, intensità. Comunque ora sono disponibili ADC economici ad un unico chip.

SOMMARIO DELLE PERIFERICHE

Per strutture specializzate esiste un certo numero di opzioni. Il seguente è un breve sommario.

1 - un sistema personale minimo ha bisogno di: microcomputer, tastiera monitor CRT (o TV), mangiacassette.

2 - un sistema migliore userà un mini floppy invece delle cassette.

3 - un sistema commerciale richiede: microcomputer con grande memoria interna, terminale CRT con tastiera commerciale, due floppy disk di grandezza piena, stampante con trasporto della carta a spinotti.

In conclusione:

- ogni sistema microcalcolatore usa una tastiera ed un CRT
- quasi tutti i sistemi usano un mangiacassette o un disk
- la maggioranza dei sistemi richiedono una stampante.

L'analisi delle caratteristiche e delle periferiche presentate in questo capitolo dovrebbe servire come linea di guida pratica importante per la scelta delle periferiche.

Nella maggioranza dei sistemi, esse costituiscono il costo maggiore.

CAPITOLO 10

SCEGLIERE UN MICROCOMPUTER

UNA MICROSTORIA DEI COMPUTERS

Il calcolo scientifico per applicazioni come l'astronomia, la navigazione, le misurazioni geografiche, e la registrazione di dati risale a migliaia di anni fa. L'antico abaco Cinese è un esempio dei primi sforzi di meccanizzare mansioni di calcolo semplici.

Il successivo evento importante sulla vista dell'automazione del calcolo fu l'addizionatore meccanico di Pascal (1643 Francia). Fu inventato dal giovane Pascal per facilitare l'esecuzione dei calcoli, ed usava ruote mobili provviste di denti, così che ogni passaggio da 0 a 9 avrebbe fatto scattare la ruota più prossima sulla sinistra (questo ora si chiama aggiungere un riporto).

Un nuovo passo concettuale fu il lavoro di Babbage sul suo "motore analitico" (dal 1820 al 1834 Gran Bretagna). Considerato da tutti come un eccentrico, Babbage non riuscì a far funzionare il proprio modello, ma, in realtà, definì un modello degli attuali calcolatori general purpose. Il suo lavoro fu in anticipo rispetto ai suoi tempi, e non fu compreso.

Il 1890 segna l'inizio dell'automazione con la scheda perforata di Hollerith, seguita dal calcolo automatico a scheda. Questo lavoro era indirizzato a risolvere il problema della tabulazione del censimento negli U.S.A. ed il prodotto segna l'entrata di una ditta conosciuta come IBM nel campo dell'elaborazione di dati.

Il prossimo passo fu l'introduzione dei primi computer alla fine della Seconda Guerra Mondiale (1944-45). Il MARK I fu sviluppato da Howard Aiken con i fondi dell'IBM, mentre l'ENIAC fu sviluppato da Eckart e Mauchly con i fondi dell'esercito degli U. S. alla Moore School dell'Università di Pennsylvania.

Il MARK I fu un "computer della prima generazione" che usava relé elettromeccanica, che funzionò dal 1944 al 1959.

L'ENIAC (Electronic Numeral Integrator and Computer) era un "computer della seconda generazione", che usava tubi a vuoto (18.000 di essi), e funzionò dal 1948 al 1958. Questa enorme macchina fu costruita per calcolare le traiettorie dei

missili. Le variazioni del programma richiedevano giorni, poiché richiedevano il ricablaggio dei fili e la risaldatura. Il computer a "programma immagazzinato" non era stato inventato ancora. L'ENIAC riempiva una grande stanza alla Moore School of Electrical Engineering, e funzionava solo per poche ore tra un'avaria e l'altra. L'ENIAC aveva solo 1K di memoria operativa (contro i 400K o più per un microcomputer contemporaneo), usava 10000 condensatori, 65000 resistenze, e 7300 relé o interruttori. Pesava 65000 libbre e occupava 3000 piedi cubici. La potenza richiesta era di 160 KW.

A questo punto l'IBM esaminò i risultati, decise che i computers, non avevano futuro, e non proseguì in questo mercato.

Nel 1945, Von Neumann formulò infine il concetto di un computer a programma immagazzinato dove sia i programmi che i dati erano immagazzinati in una memoria permettendo una generalità completa nell'eseguire qualsiasi programma. L'ENIAC fu quindi seguito da una certa quantità di --- AC's (Automatic Computers):

- EDVAC (Electronic Discrete Variable Automatic Computer)
- EDSAC (Electronic Delay Storage Automatic Computer)
- UNIVAC (Universal Automatic Computer) creato nel 1951 da Eckart e Mauchly per l'ufficio dei censimenti degli U.S. L'IBM allora comprese di aver fatto una valutazione errata del mercato, e riversò tutto il proprio peso nell'impossessarsene. Ora il lettore stabilirà quanto bene essi vi siano riusciti.

L'IBM introduce il 701 nel 1953, e si impadronisce di una fetta vasta del mercato.

I transistor diventano una realtà alla fine degli anni cinquanta e permetteranno la nascita di computer della "terza generazione".

Il 1964-66 è marcato da eventi importanti.

L'IBM introduce la serie 360 che conferirà all'azienda il suo predominio mondiale.

La DEC (Digital Equipment Corporation) introduce il PDP 6, seguito dal PDP 8, il minicomputer di maggior successo.

Dall'altra parte della gamma, la CDC (Control Data Corporation) introduce il CDC 6600, un "super computer" per i calcoli numerici più complessi.

La quarta generazione

All'inizio degli anni 60 l'area attorno Sunnyvale, all'estremità Sud della Baia di San Francisco era ancora una pacifica area agricola, conosciuta per i suoi frutteti. È ora diventata un esteso parco industriale conosciuto come la "*Silicon Valley*" (Valle del Silicio). Qui è dove si è insediata la maggioranza dei costruttori di semiconduttori. Attraverso gli anni sessanta, divenne possibile integrare un numero sempre maggiore di transistor, ed altri componenti, su di un singolo chip di silicio, finché, all'inizio degli anni 70 si poterono realizzare su un chip diverse centinaia di

transistor: questa era l'era dell'LSI (Large Scale Integration). Una tale quantità di transistor rende possibile realizzare un computer semplificato, o piuttosto la sua CPU (vedi il capitolo 3) su un chip.

Nel Novembre 1971 fu annunciato dalla Intel il 4004, il primo microprocessore; a quel tempo la Intel era una piccola ditta nella Silicon Valley. Ci volle più di un anno perché la Intel ed altri costruttori comprendessero che per sbaglio era stato introdotto un nuovo componente rivoluzionario. La maggior parte dei microprocessori principali conosciuti al giorno d'oggi furono introdotti allora: l'8080 della Intel, il 6800 della Motorola, lo Z80 della Zilog, il 6502 della MOS Technology, ecc.

Tuttavia tutte le applicazioni dei microprocessori erano nel campo dei computer, nel controllo industriale,o in aviazione. Proprio come i computer, al loro inizio, essi furono usati per scopi "scientifici", e non ancora per l'elaborazione dei dati. Nessuno aveva ancora compreso la loro potenzialità.

Il seguente evento significativo accadde nel Gennaio 1975, con l'annuncio del personal computer ALTAIR della MITS, che era allora una piccola azienda del Nuovo Messico. Durante gli anni seguenti spuntarono in tutta la nazione fabbricanti di microcomputer per soddisfare questo nuovo mercato. La maggior parte non poterono distribuire abbastanza elementi, ed alcuni non riescono ancora adesso. In questo capitolo è presentata un'indagine dettagliata di prodotti disponibili.

MICROCOMPUTERS COMMERCIALI

Nei capitoli precedenti sono stati presentati i criteri di selezione per la scelta della CPU di un microcomputer e delle periferiche. Essi possono ora essere applicati a sistemi esistenti, così da valutare i loro vantaggi e svantaggi.

Finora sono stati differenziati due principali tipi di formati di assemblaggio:

1 - il microcomputer a boards singolo
2 - il sistema microcomputer
Ora si può distinguere un terzo formato di assemblaggio
3 - il microcomputer a "tastiera integrata".

Il microcomputer a tastiera integrata include la tastiera ed il microcomputer nella stessa scatola. Questo è l'assemblaggio usato per sistemi economici ad uso domestico. I vantaggi sono la facilità di maneggiarlo, minore costo del contenitore, buon aspetto. Lo svantaggio è di solito la mancanza di spazio all'interno del contenitore per una futura espansione (boards addizionali di interfaccia e di memoria).

Ora saranno studiate le tre categorie di cui sopra.

Attenzione: a causa dell'alto ritmo di introduzione di nuovi sistemi (ed il ritiro dal mercato di alcuni di essi) questa non ha la pretesa di essere un'indagine completa. Anche i prezzi indicati possono cambiare rapidamente, come pure le caratteristiche tecniche. L'analisi viene proposta per il suo valore educativo.

MICROCOMPUTERS A SINGOLA SCHEDA

Le limitazioni di un microcomputer a scheda singola, sono già state messe in rilievo. Al di fuori del campo ingegneristico e di controllo, il loro valore essenziale sta nel loro pregio educativo nell'imparare tecniche elementari di programmazione o di interfacciamento. La maggior parte dei costruttori di microprocessori offrono un microcomputer a scheda singola equipaggiato con una tastiera minima e quattro o sei LED a costo minimo che possono essere usati per tali scopi.

Vari microcomputer single board sono stati progettati specificatamente come strumenti didattici a basso costo, con tastiera on-board a LED. Essi meritano una breve menzione qui.

IL KIM-1

Introdotto originariamente dalla MOS Technology (ora è una divisione della Commodore), il KIM-1 è anche disponibile da parte della Rockwell International. È stato anche annunciato un "KIM-1 migliorato" come VIM-1 dalla Synertek (una seconda fonte del microprocessore 6502 usato nel KIM-1).

Il KIM-1 ha sul board: 1K di RAM, due combinazioni I/O 6530 (=2K di ROM), una tastiera a 23 tasti, sei LED, come pure le interfacce per una teletype, e per un mangiacassette standard.

Per usare il KIM, uno ha semplicemente bisogno dell'alimentazione. Per collegarsi ad un mangiacassette si ha bisogno appena di collegare l'I N audio e l'OUT audio. Il collegamento ad una teletype è altrettanto semplice.

Il NEC TK80

Il TK-80, un compatto microcomputer a scheda singola, è stato progettato come strumento didattico per conoscere l'8080A (costruito pure dalla NEC in Giappone). È equipaggiato con una tastiera esadecimale, più nove tasti per comandi di controllo, ed otto LED per il display.

Una porzione del board è lasciata libera per il breadboarding (esercitazioni sperimentali con i componenti) dell'utente. La ROM/PROM e la RAM sono espandibili fino a 1K byte sul board. Il programma monitor (di gestione) risiede negli indirizzi di memoria ROM da 0000 a 02FF (755 byte). È disponibile una batteria tampone per la RAM.

Variazioni/Imitazioni

Come molti altri prodotti, il board della NEC è stato risiglato da alcune ditte con le loro proprie sigle, dopo aver ricodificato i tasti, ed installato un programma monitor modificato. È tuttavia il board originale della NEC, con lo stesso hardware.

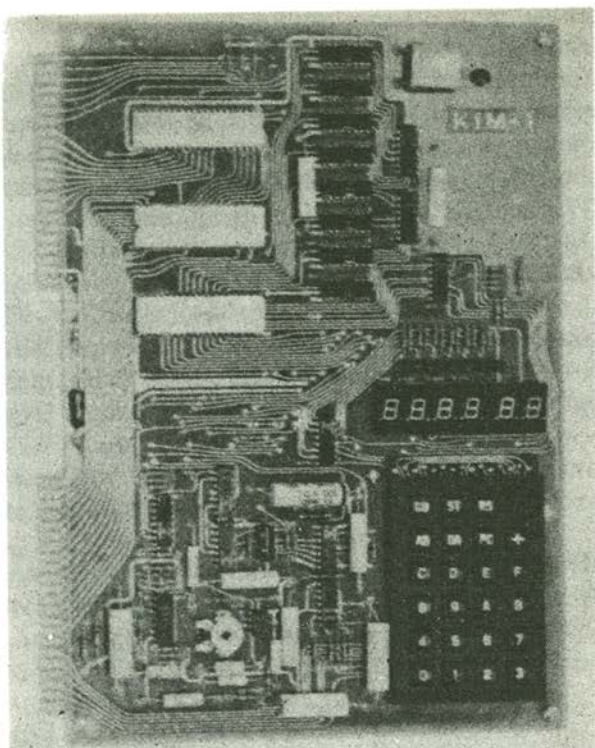



Fig. 10-1: Il Microcomputer a singola scheda KIM-1.

In modo simile, sono disponibili pannelli o contenitori su misura per un gran numero di microprocessori così da "personalizzarli" alle descrizioni della ditta acquirente.

Sistemi Home a costo minimo

I sistemi a costo minimo sono quelli a meno di 1000 dollari per un sistema completo ed adoperabile, e possono essere veramente comperati negli Stati Uniti, per un costo dai 500 dollari in su. Ora li esamineremo.

Il Videobrain della Umtech

Caratteristiche:

- *Microprocessore usato: l'"F8" della Fairchild (3870)*
- *Memoria standard: 4K di ROM, 1K di RAM*

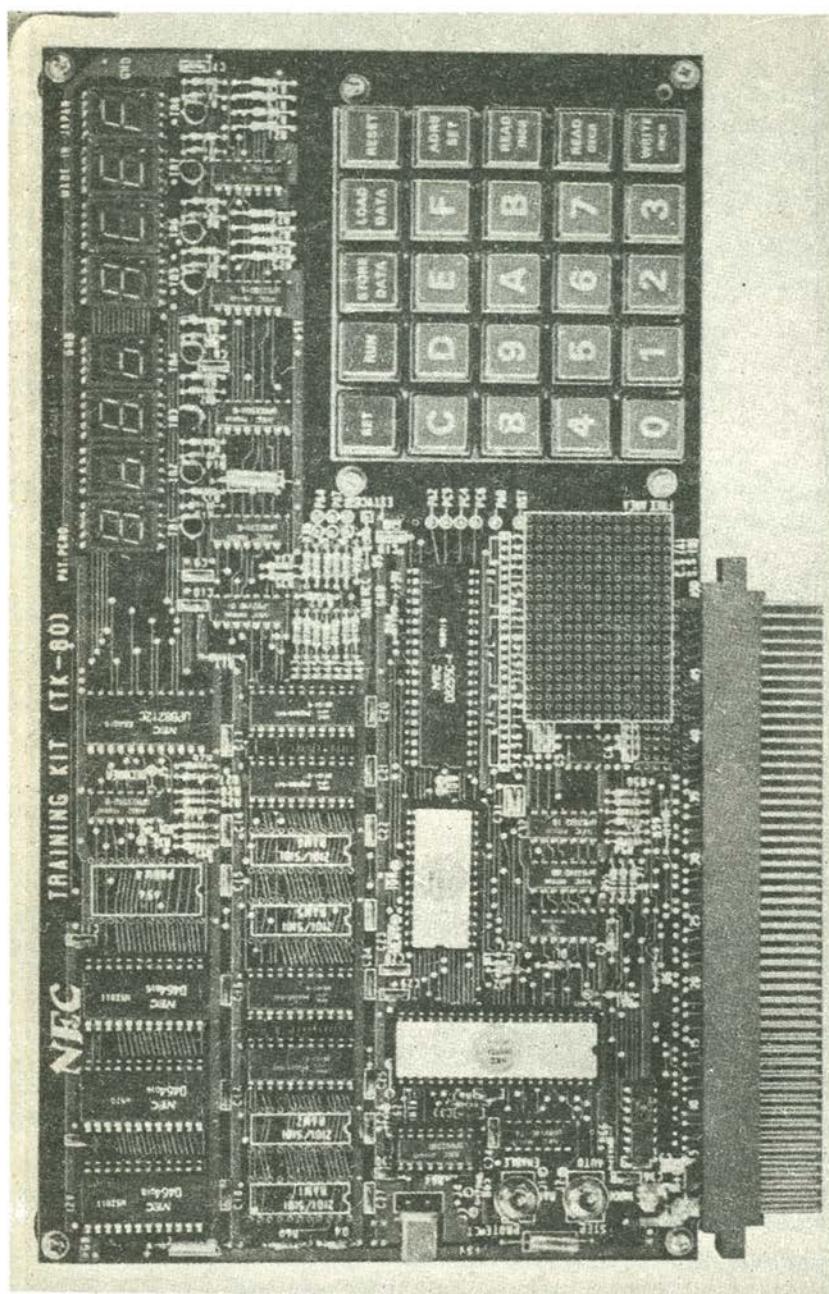



Fig. 10-2: Il Microcomputer a singola scheda della Nippon Elettric.

Fig. 10-3: Un "kit" (della Motorola).

- *Tastiera: 36 tasti, 71 caratteri*
- *I/O: uscita per TV a colori (Radio Frequenza). Connessioni RS 232 e current loop.*

Il Videobrain è un computer con "tastiera integrata" progettato come un gradino sopra ai giochi televisivi. Usa "cartucce di programma" che forniscono fino a 4K di ROM, ed è provvisto di paddles, e di una switch box di antenna per essere collegato ad un televisore a colori domestico.

È attualmente distribuito da Macy's una catena di grandi magazzini USA, per 500 dollari.

Questo computer è progettato esclusivamente per l'utente che non vuole programmare, e vuole solamente inserire un nuovo programma. L'utente dipende completamente dal costruttore per la fornitura di nuovi programmi, ed il computer non può essere immediatamente esteso al livello di processore.

Comunque, poiché la maggioranza degli utenti di home computer non vuole imparare a programmare, esso può essere ben accetto se diventano disponibili un numero sufficiente di cartucce di programma (program cartridges).

Gli usi tipici: giochi TV, tenuta della cassa di casa, analisi dei prestiti, insegnante di matematica, insegnante di musica, corso di dattilografia al tatto (senza guardare i tasti della macchina).

Professional Arcade della Bally

Caratteristiche:

- *Microprocessore usato: lo Z80 della Zilog*
- *Memoria standard: 8K di ROM, 4K di RAM*
- *Tastiera: 24 tasti, 70 caratteri*
- *I/O: uscita (RF) per TV a colori. Effettua un display di 11 linee × 27 caratteri.*
- *Memoria addizionale: cassette di 8K byte di ROM. Espandibile fino a 44K con la scatola di espansione che ha come caratteristiche 16K di ROM. 16K di RAM, tastiera alfanumerica standard, interfaccia IEEE 488.*
- *Prezzo in USA: 300 dollari*

Progettata come "super gioco video", questa unità ha come caratteristica una tastiera etichettata in mini-BASIC, come pure in caratteri ordinari. I programmi sono immagazzinati in cassette ROM, e l'unità include l'immagazzinamento per 15 cassette. È stata annunciata una scatola di espansione, che dovrebbe trasformare l'unità in un computer general purpose completo, con una tastiera alfanumerica standard.

Usi tipici: giochi TV, insegnante di matematica, finanza domestica.

Il PET della Commodore

Caratteristiche:

- *Microprocessore usato: il 6502*
- *Memoria standard: 8K di RAM, 14K di ROM (comprende 8K di BASIC)*

Fig. 10-4: Il videobrain.

- *Tastiera: 73 tasti, 128 caratteri*
- *Monitor CRT: da 9 pollici integrato, bianco e nero, 25 linee × 40 caratteri, matrice di punti 8 × 8.*
- *Mangiacassette: integrato*
- *I/O: interfaccia standard IEEE 488*
- *Caratteristiche particolari: limitata capacità di disegno (64 segni grafici)*
- *Prezzo in USA: 800 dollari con 8K di RAM*

Fig. 10-5: Il Bally Arcade usa cartucce di programma.

PET sta per "Personal Electronic Transactor" (Operatore Elettronico Personale) ed è un sistema completo, ed integrato, con un proprio display, ed una memoria di massa (un mangiacassette). I programmi possono essere caricati dalle cassette, o, in futuro, da altre periferiche attraverso l'interfaccia IEEE 488. Eccetto che per una quantità limitata di memoria RAM interna, che limita la portata dei programmi, è un computer general purpose completo. È equipaggiato con una tastiera alfanumerica completa che presenta 64 caratteri grafici (vedi l'illustrazione), ma che ha lette-

Fig. 10-6: Il PET é un'unità integrata.

re maiuscole e minuscole (tuttavia i segni grafici e le minuscole si escludono mutuamente).

La ROM di 14K interna include: un'interprete BASIC (8K), un sistema di funzionamento (4K), una diagnostic routine (1K), un monitor in linguaggio macchina (1K). L'interprete BASIC è piuttosto completo poiché fornisce files aritmetici a 10 cifre con virgola fluttuante ed array multidimensionali, come pure il character addressing (indirizzo a caratteri comandi PEEK e POKE).

Uno svantaggio possibile è la mancanza di un connettore RS 232, che limita il numero di periferiche che possono essere facilmente collegate all'unità.

Usi tipici: giochi, didattica, finanza domestica, applicazioni commerciali semplificate.

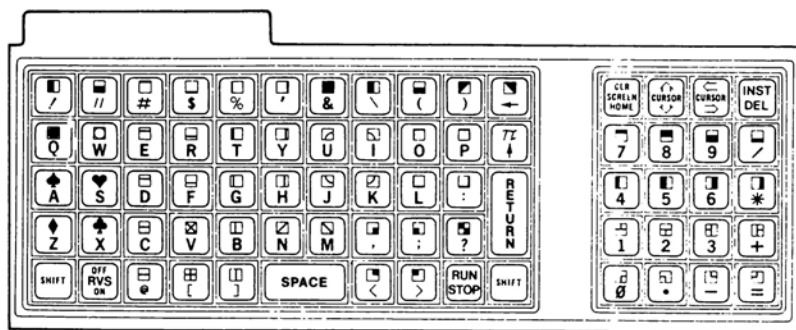



Fig. 10-7: La tastiera del PET presenta caratteri grafici.

Il TRS-80 della Radio-Shack

Caratteristiche:

- *Microprocessore usato: Z-80*
- *Memoria Standard: 4K di ROM, 4K di RAM. Espandibile internamente fino a 12K di ROM e 16K di RAM.*
- *Tastiera: 53 tasti, stile macchina da scrivere.*
- *Display: video da 12 pollici facente parte del sistema. 16 linee × per 64 caratteri. segni grafici 128 orizzontali per 48 verticali.*
- *I/O: mangiacassette facente parte del sistema (250 bpo, non c'é correzione degli errori). Port di espansione.*
- *Prezzo in USA: 600 dollari (con 4K di ROM, mini BASIC di livello I).*

È un competitore diretto del PET, è pure esso un sistema completo con tastiera (staccata), monitor CRT, e mangiacassette. Le nuove periferiche annunciate com-

prendono i floppy disk, e la stampante, che devono essere collegati attraverso il port di espansione appropriato, sul retro della tastiera o del contenitore del computer. Il nuovo BASIC di "livello II" è un BASIC normale e completo con la possibilità di usare un assembler, per programmare in linguaggio macchina. I programmi di solito vengono caricati sul sistema dalle cassette.

La sua tastiera è una tastiera "QWERTY" standard da macchina da scrivere, attraverso la quale vengono introdotti i comandi.

Fig. 10-8: Il TR5-80 è completo ma scomponibile.

È in progetto una scatola adottatrice per S-100, che dovrebbe permettere la connessione al sistema di board S-100 (esternamente). C'è ora sul retro della tastiera un connettore speciale con l'estremità a 40 pin.

Usi tipici: giochi, didattica, applicazioni commerciali limitate (libro paga), cucina, finanza personale.

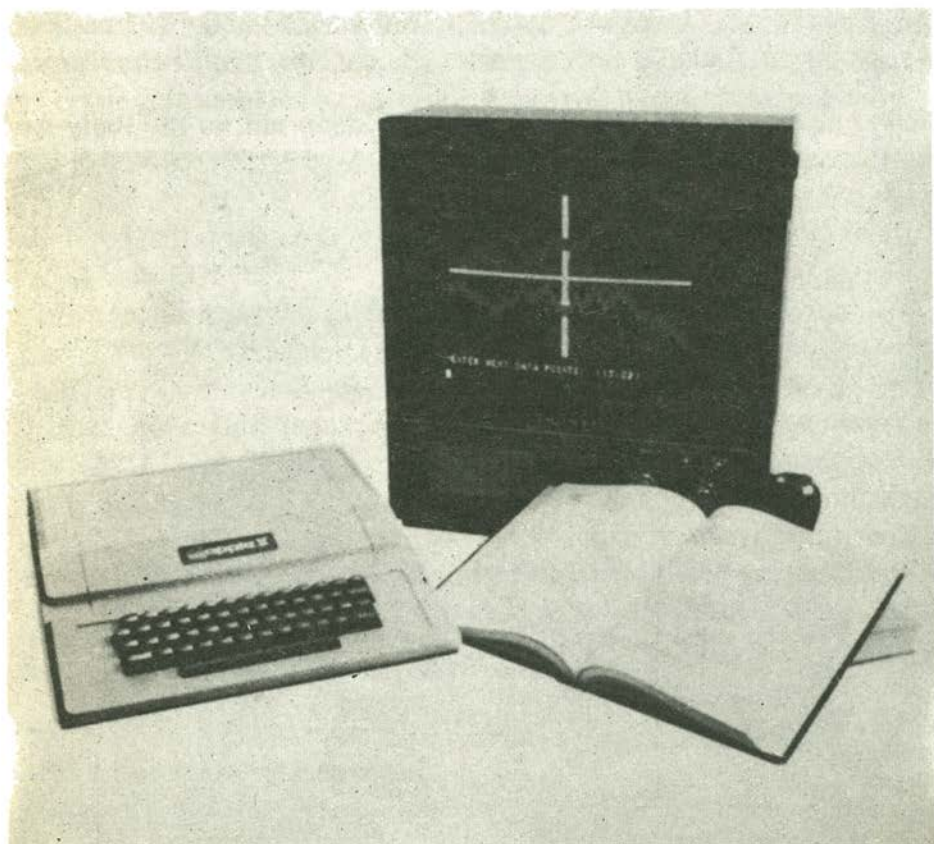



Fig. 10-9: L'Apple ha una tastiera integrata e si collega ad un televisore a colori.

MICROCOMPUTERS GENERAL PURPOSE

Apple II

Caratteristiche:

- *Microprocessore usato: 6502*
- *Memoria standard: 8K di ROM (BASIC incluso), 4K di RAM, espandibile internamente fino a 48K di RAM*
- *Tastiera: 52 tasti, stile macchina da scrivere*
- *I/O: uscita (RF) per TV a colori 24 righe × 40 caratteri (caratteri maiuscoli 5 × 7). Capacità di segni grafici: 280 orizzontali per 192 verticali, in quattro colori. Comprende l'interfaccia per mangiacassette, altoparlante, connettori (paddle).*

In apparenza, questa unità è vicina al Videobrain, o al Bally Arcade, a causa della tastiera integrata. Tuttavia, esso offre la possibilità di espansione all'interno della scatola fino a 48K byte di RAM (sul board singolo), e due prese per ROM addizionale. Come risultato, ha una capacità di computing completa, senza alcuna limitazione di memoria.

Richiede un televisore esterno, e contiene l'interfaccia per il TV colori, come pure un'interfaccia standard per mangiacassette, ed interfacce paddle.

Gli 8K di ROM includono un "monitor" di 2K (che comprende un miniassembler, disassembler, debugger, package per virgola fluttuante) più 6K di BASIC con alcune limitazioni (numeri interi solo da −32767 a +32767 ed array a singola dimensione).

I caratteri sono limitati alle maiuscole.

I programmi di solito vengono introdotti prelevandoli dalle cassette, letti attraverso un registratore standard.

Usi tipici: giochi TV a colori, computing general purpose.

Altair 8800-b

Caratteristiche:

- *Microprocessore usato: 8080*
- *Memoria standard: qualsiasi quantità di ROM/RAM fino a 64K*
- *I/O: RS - 232, 20 mA current loop. Port I/O seriale*
- *Particolarità: un pannello frontale (opzionale). 16 board addizionali S-100 che possono stare nella scatola di base.*

L'Altair fu, storicamente, il primo microcomputer introdotto, ed è un computer general purpose. Esso introdusse il bus S-100 che ora è un bus standard. La scatola base può essere scelta dal cliente per quanto riguarda il contenuto delle schede (board). È disponibile una varietà di board: memorie, interfaccia, special-purpose.

Per essere usata, questa scatola a microcomputer deve essere collegata ad un ter-

minale CRT (con la tastiera), ad una memoria di massa (come un disk) e come optimum, ad una stampante.

Ha del software di estensione a disposizione: DOS, BASIC in time sharing, (per coordinare il funzionamento dei vari elementi in base ai diversi tempi operativi), Assembler Editor, programmi Commerciali. La sua versione 4.0 BASIC è diventata un riferimento standard (non necessariamente la migliore, ma standard come completezza).

Per l'utente commerciale: il software è ora disponibile attraverso una ditta a parte: Altair, Software Distribution Company, 27111 Erwin St., Woodland Hills, CA 91367. Gli attuali packages commerciali includono il word processing, registro generale, inventario, libro paga, ricevibili, pagabili.

Fig. 10-10: Il sistema Altair in una delle numerose versioni.

Imsai 8080

Caratteristiche:

- *Microprocessore usato: 8080A*
- *Memoria: qualsiasi quantità di ROM/RAM fino a 64K*
- *I/O: dipende dal (o dai) board di interfaccia scelti*
- *Particolarità: un motherboard (board madre) S-100 (22 parti aggiungibili). Pan-pannello frontale.*

L'Imsai è un competitore diretto dell'Altair, introdotto poco dopo quest'ultimo. Usa il bus S-100, così che anch'esso può accogliere una varietà di board S-100.

Le sue caratteristiche sono essenzialmente analoghe al microcomputer precedente.

Esso "di natura" ha un'interprete BASIC e può eseguire il software dell'8080, così che i packages scritti in codice 8080 funzioneranno pure sull'Imsai.

Per l'utente commerciale: è stata scritta una varietà di packages per l'IMSAI o per l'ALTAIR. Essi non sono compatibili, se scritti in BASIC. Dovete verificare in che BASIC sono scritti, cioé con che interprete possono funzionare. È vero che ogni programma BASIC può essere adottato ad un altro interprete (purché l'interprete sia altrettanto completo), tuttavia, è un compito tedioso, suscettibile di errori, ed è un dispendio di tempo.

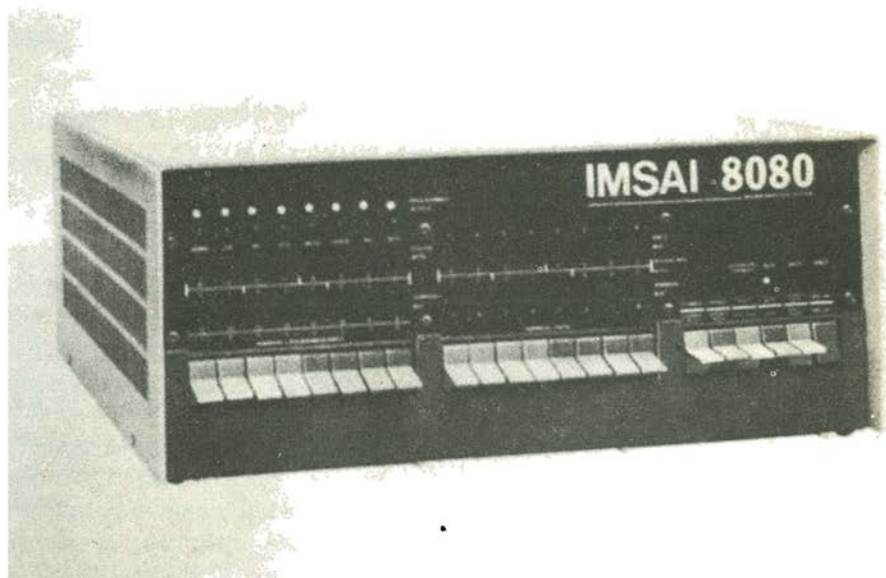



Fig. 10-11: L'IMSAI tradizionale.

Fig. 10-12: "Business Packaging" della IMSAI.

Processor Technology Sol-20

Caratteristiche

- *Microprocessore usato: 8080*
- *Memoria: 1K di PROM (espandibile a 2K), 2K di RAM (espandibili a 64K)*
- *Tastiera: del tipo standard da macchina da scrivere, più tastiera numerica integrata (85 tasti). Maiuscole e minuscole.*
- *I/O: uscita video (necessità di un monitor video, o TV **adattata** - vedi il capitolo sulle periferiche per la differenza da un TV standard). Comprende un'interfaccia per audio cassette(1200 bps).Connettori (RS-232 e 20 mA current loop) paralleli e seriali.*
- *Particolarità: un bus S-100 standard. La possibilità di aggiungere cinque moduli di espansione.*

Con la sua tastiera integrata, questo computer assomiglia ad un sistema domestico a costo minimo. Tuttavia, esso ha una tastiera completa di tipo commerciale, ed ha abbastanza spazio interno per un sistema completo a 64K. Perciò è funzionalmente equivalente all'IMSAI o all'ALTAIR. In aggiunta esso include su di un board singolo la maggior parte delle interfacce desiderabili in un tale sistema (eccetto che per i disk) senza la necessità di inserire boards addizionali.

Dal lato negativo, possono essere inseriti solo cinque moduli addizionali, il che è una possibile limitazione per l'utente che prevede di usare una grande quantità di opzioni. Comunque, è anche improbabile che la maggioranza degli utenti abbiano bisogno di altre parti.

Naturalmente, è disponibile un BASIC di 8K, come pure altri packages usuali.

A causa del bus S-100, la maggior parte dei moduli disponibili dall'IMSAI e dell'ALTAIR possono essere usati nel SOL.

Venendo dopo l'ALTAIR e l'IMSAI, questa unità è costruita completamente priva di pannello frontale, ed il progetto globale comporta un assemblaggio altamente compatto (interruttori DIP-dual in line package simile ai componenti sono disponibili internamente per i nostalgici del pannello frontale).

Cromemco Z2

Caratteristiche:

- *Microprocessore usato: Z-80 (4MHz)*
- *Memoria: 1K di PROM, 32K di RAM (2 board da 16K), espandibile*
- *I/O: a seconda di board di interfaccia scelti. RS-232 a current loop, più interfaccia parallelo ad 8 bit.*
- *Particolarità: usa il bus S-100. 21 board aggiuntivi. Disponibile in scatola integrata con 1 o 2 disk drivers.*

Questo microcomputer appartiene alla stessa famiglia di prodotti compatibili con l'S-100. Il microprocessore scelto è più veloce dello Z-80 standard. Comunque (vedi il capitolo "Scegliere un Sistema"), questo non risulta necessariamente migliore del rendimento globale. Devono essere esaminati i tempi di esecuzione del software su di un sistema equipaggiato con disk per ogni package. Con un package ben progettato il sistema può migliorare le sue prestazioni. Il supporto software comprende un BASIC di 16K ampliato, un compilatore FORTRAN IV. È anche disponibile un pannello frontale in pratica identico a quello dell'IMSAI 8080 opzionale. Un "dazzler" (creatore di effetti luminosi) a due board fa da interfaccia fra il sistema ed un TV a colori standard (richiede 2K di memoria). È anche disponibile una console provvista di joystick e di altoparlante.

North Star Horizon

Caratteristiche:

- *Microprocessore usato: Z-80 a 4 MHz*

Fig. 10-13: Il SOL ha una tastiera integrata "terminal type".

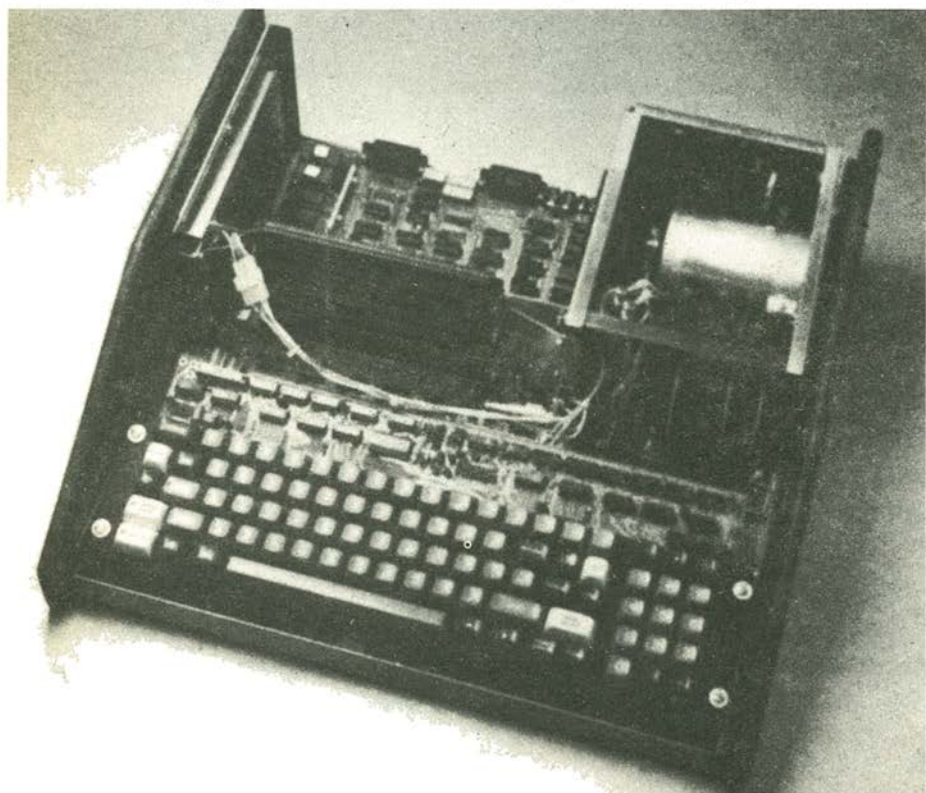



Fig. 10-14: Rimosso il coperchio, si vede l'elettronica della tastiera (di fronte) l'alimentazione (a destra), e il board S-100 del microcomputer.

- *Memoria: espandibile fino a 64K*
- *Particolarità: disk drive doppio integrato, Bus S-100*

Questa unità è pure compatibile con l'S-100, ha un BASIC buono e completo e usa la versione veloce dello Z-80. È disponibile un board di virgola mobile per una aritmetica veloce.

Parasitic Engineering Equinox

Caratteristiche:

- *Microprocessore usato: 8080*
- *Particolarità: bus S-100, board speciale per pannello frontale. Il pannello frontale ha una tastiera ottale e dei LED, con single-step.*

Fig. 10-15: Lo Z-2 con l'opzione a doppio disk.

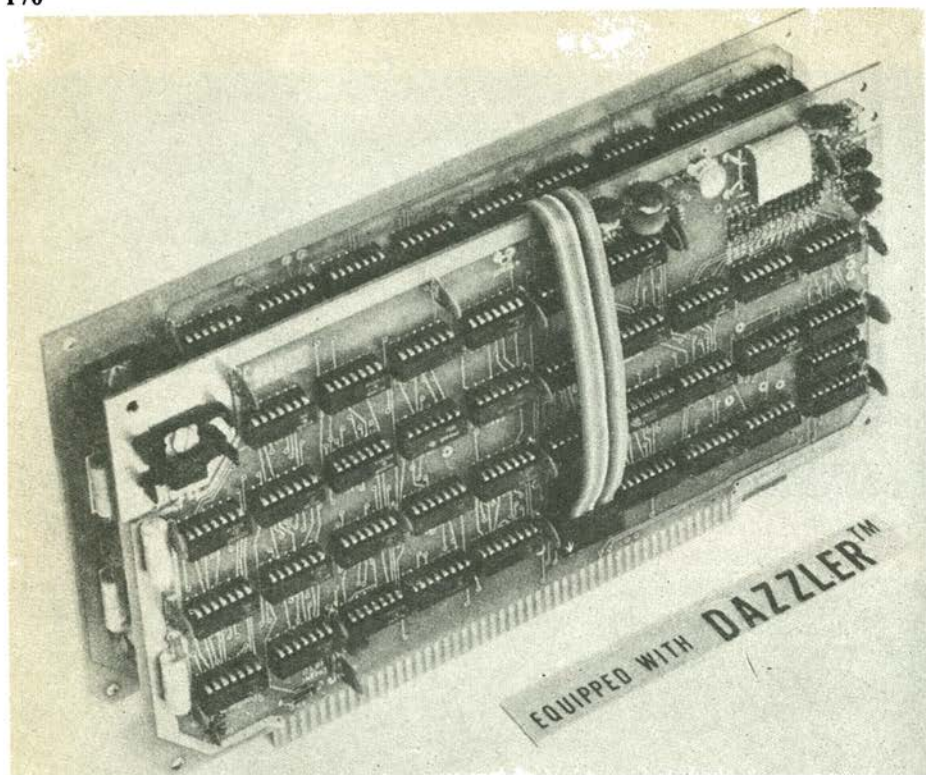



Fig. 10-16: Il Dazzler usa due piastre e fornisce l'interfaccia per il televisore a colori.

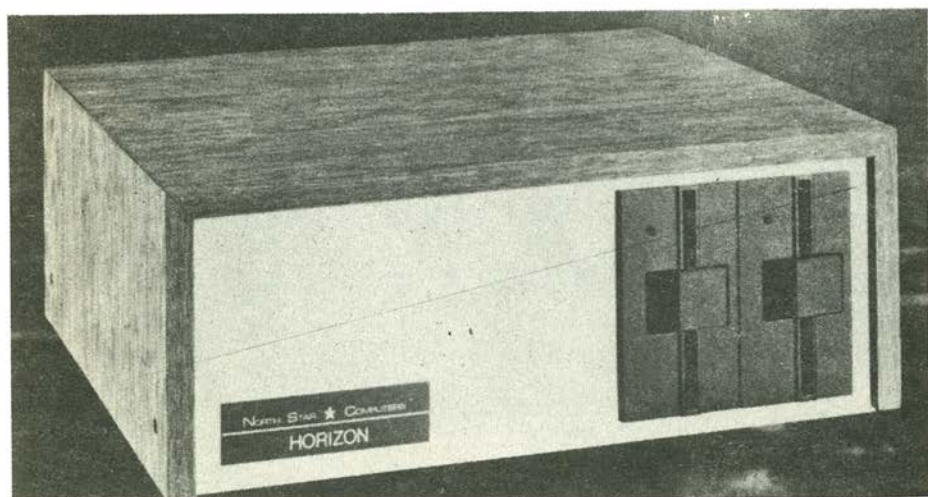



Fig. 10-17: North Star Horizon.

Technical Design Labs Xitan

Caratteristiche:
- *Microprocessore usato: Z-80*
- *Memoria espandibile fino a 64K*
- *I/O: due port seriali, interfaccia per cassette da 1200 baud*
- *Particolarità: bus S-100*

Ohio Scientific Challenger

La Ohio Scientifci offre una gamma di sistemi che usano il 6502, lo Z-80 ed il 6800 su di un board singolo, come pure una varietà di board di interfaccia. È anche disponibile un software esauriente.

Vector Graphic Vector 1

Caratteristiche:
- *Microprocessore usato: 8080*
- *Memoria: espandibile fino a 64K*
- *Particolarità: bus S-100*

Fig. 10-18: Il Vector 1.

Polymorphic 88

Caratteristiche:

- *Microprocessore usato: 8080*
- *Memoria: 3K di ROM, 16K di RAM, espandibile fino a 64K.*
- *Particolarità: bus S-100 (5 parti aggiungibili). Varie opzioni di package disponibili.*

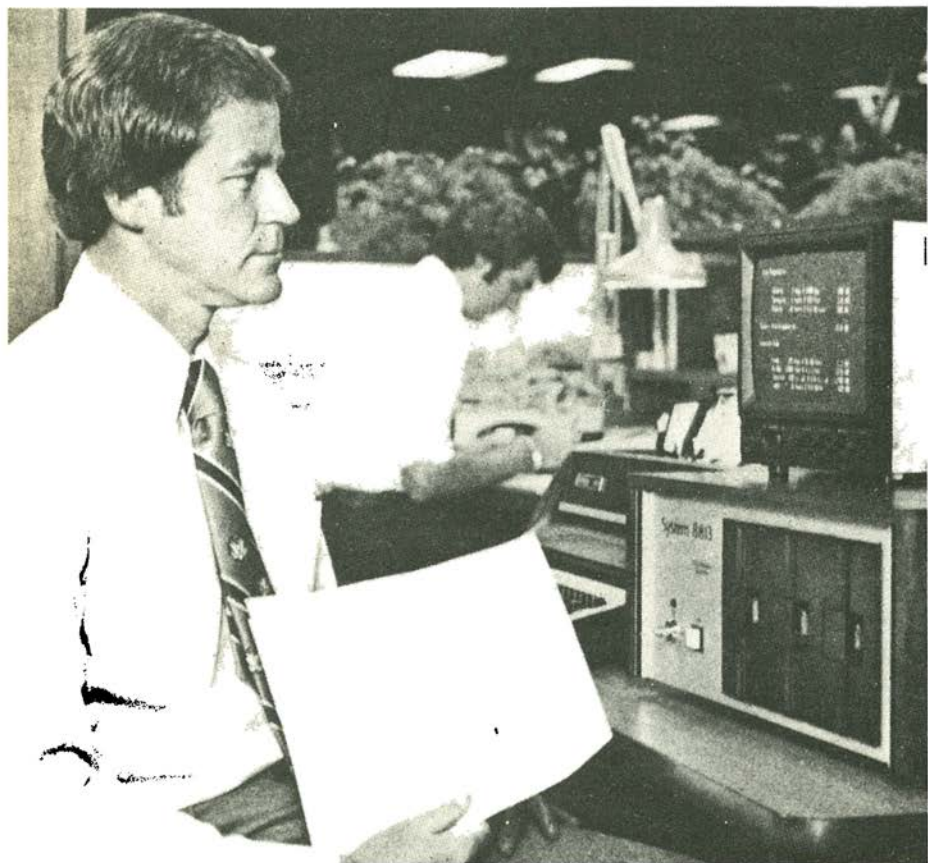



Fig. 10-19.

Heathkit H8

Caratteristiche:
- *Microprocessore usato: 8080*
- *Memoria: 4K di RAM, espandibili fino a 16K sulla scheda (il board)*
- *Particolarità: tastiera esadecimale + LED. (Ora è compatibile con l'S-100)*

L'H8 è l'estremità inferiore dell'entrata dell'Heathkit nel mercato dei microcomputers. E' disponibile una varietà di periferische che si collegano facilmente al sistema.

Fig. 10-20: L'Heathkit H8.

Heathkit H11

Caratteristiche:
- *Microprocessore usato: LSI 11*
- *Memoria: completamente espandibile fino a 64K*
- *Particolarità: usa il Q bus standard LS11. Può usare il software del PDP 11, su licenza della Digital Equipment Corporation.*

Una delle deficienze importanti dei microcomputers è stata la mancanza di una vasta libreria di software adoperabile. Il PDP11 è uno dei microcomputer per i quali esiste la più vasta libreria software. L'H11 rappresenta un tentativo di trarre vantaggio da ciò. Usa internamente il set di chip LSI 11, che emula un PDP 11/03 cioè accetta lo stesso insieme di istruzioni: i programmi del PDP 11/03 possono essere eseguiti sull'H11. In aggiunta, possono essere collegate facilmente le periferiche LSI 11 standard.

Una parola di avvertimento: il "Q-bus" dell'LSI 11 non è identico all'"Unibus" del PDP11.

Questo sistema dovrebbe interessare tutti quelli che hanno usato o che hanno accesso alla libreria del PDP11.

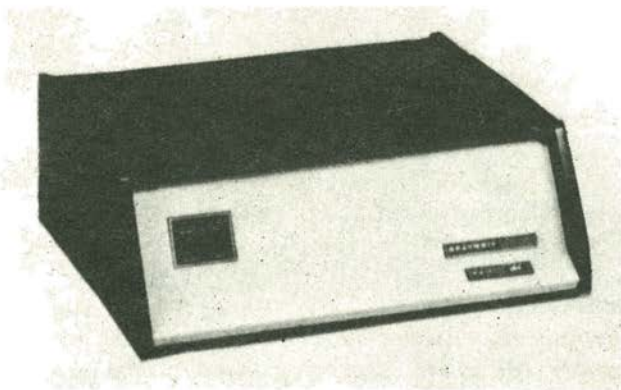



Fig. 10-21: L'Heathkit 11.

Sord

Caratteristiche:
- *Microprocessore usato: Z-80*
- *Memoria: espandibile fino a 64K ed oltre*

- *I/O: display CRT da 12 pollici integrato (1920 caratteri), stampante a 40 colonne built-in (incorporata), mini-floppy drive, ed ampia tastiera.*
- *Particolarità: sistema completamente integrato. Bus S100*

Questo sistema completamente integrato è una delle prime introduzioni giapponesi nel settore.

Fig. 10-22: Il giapponese SORD.

Digital Group Bytemaster

Caratteristiche:

- *Microprocessore usato: Z-80*
- *Memoria: da 18K a 64K, a seconda della mainframe (struttura principale - cioé il tipo di microcomputer).*
- *I/O: tastiera completa fornita. CRT integrato (9 pollici, 16 linee × 94 caratteri, maiuscole e minuscole in completo). Mini floppy integrale a doppia faccia (o cassetta digitale).*

Questo è un sistema di piccole dimensioni, e completamente integrato con la tastiera separabile, in modello da tavolo. La struttura principale può anche sostenere fino a 64K di memoria che sono disponibili in moduli separati, e sono pure disponibili periferiche esterne addizionali. Tutte le periferiche hanno schede di interfaccia che vengono inserite nella appropriata fessura praticata nella struttura principale. Non è un sistema S-100.

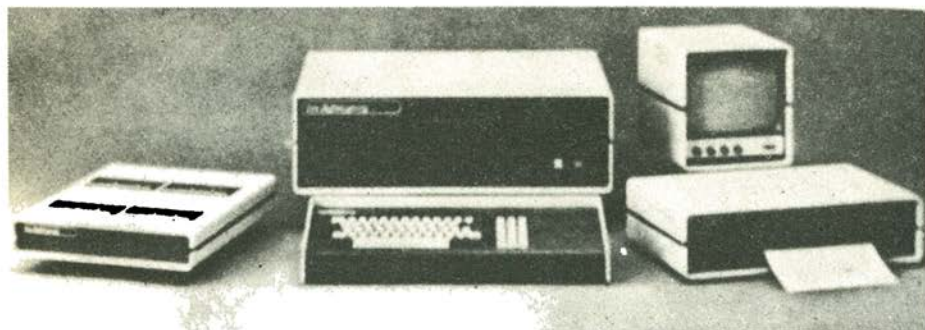



Fig. 10-23: Digital Group System.

Compucolor II

Caratteristiche:

- *Microprocessore usato: 8080A*
- *Memoria: 16K di ROM, 8K RAM (comprende 4K RAM rinfrescabile), espandibile fino a 32K di ROM, 32K di RAM.*
- *I/O:CRT a colori integrato (13 pollici), 32 righe × 64 caratteri tastiera standard ASCII opzionale. Mini-floppy integrato, Part RS-232C.*
- *Particolarità: Vector software, 64 caratteri speciali.*

Questo sistema fornisce sia il general purpose computing che grafici a colori limitati.

Digital Equipment PDP 11/03

Caratteristiche:

- *Microprocessore usato: insieme dei chip LSI11*
- *Memoria: 4K, espandibile fino a 64K.*
- *Particolarità: compatibile all'insieme di istruzioni del PDP11. Software eccellente. Fino a sei moduli addizionali nel contenitore.*

Questo microcomputer fu sviluppato dalla DEC come sostituzione a costo minore del minicomputer 11/05, ed usava originariamente il set di chip della Western Digital (l'LSI 11). Questo sistema è adoperabile come personal computer ma è stato essenzialmente venduto sulla linea di vendita principale della DEC cioé sui mercati del laboratorio e del controllo.

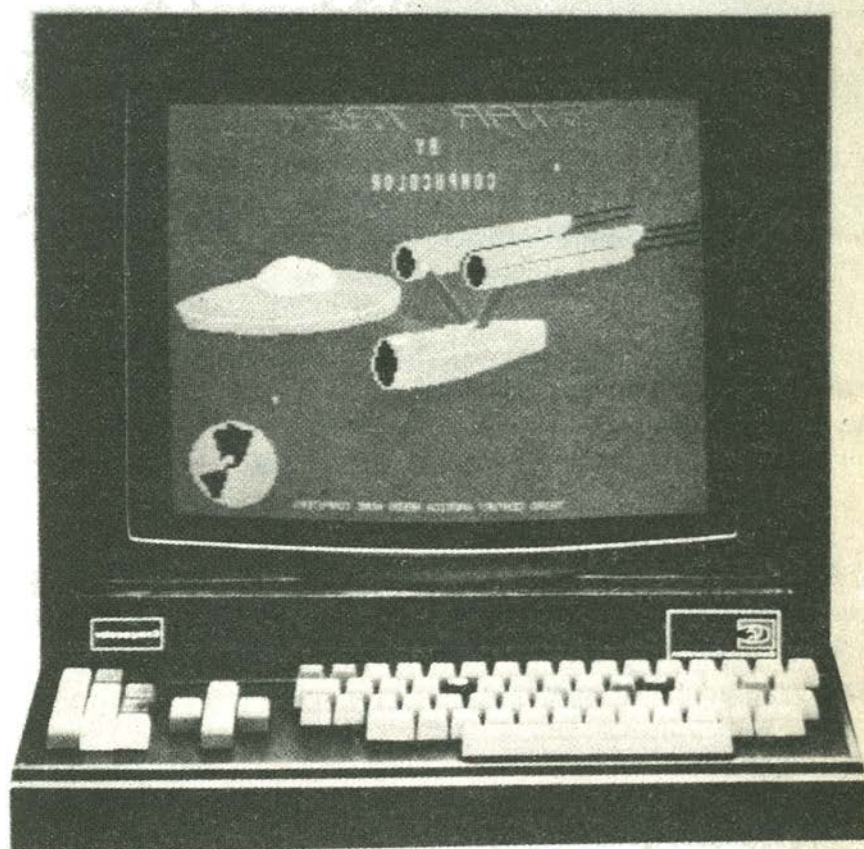



Fig. 10-24: Il Compucolor ha uno schermo a colori integrato. Questo modello ha il disco separato.

Come tutta la linea PDP11, esso ha un ampio software, in particolare Dartmouth BASIC, ANSI, FORTRAN, e RT-11, un sistema funzionante nel tempo reale con operazioni di foreground/background.

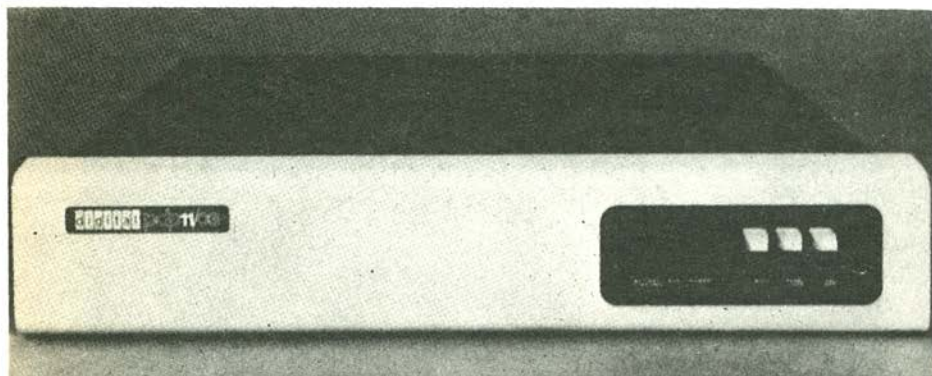



Fig. 10-25: Il PDP 11/03 della Digital.

South West Technical Products SWTC 68/2

Sistema integrato con tastiera stile macchina da scrivere e display CRT. Usa il microprocessore 6800, e per esso ha definito il bus SS-50.

Alpha-Micro Systems

Questo microcomputer offre il processing a 16 bit usando il processore WD-16 della Western Digital. Compatibile col bus S-100.

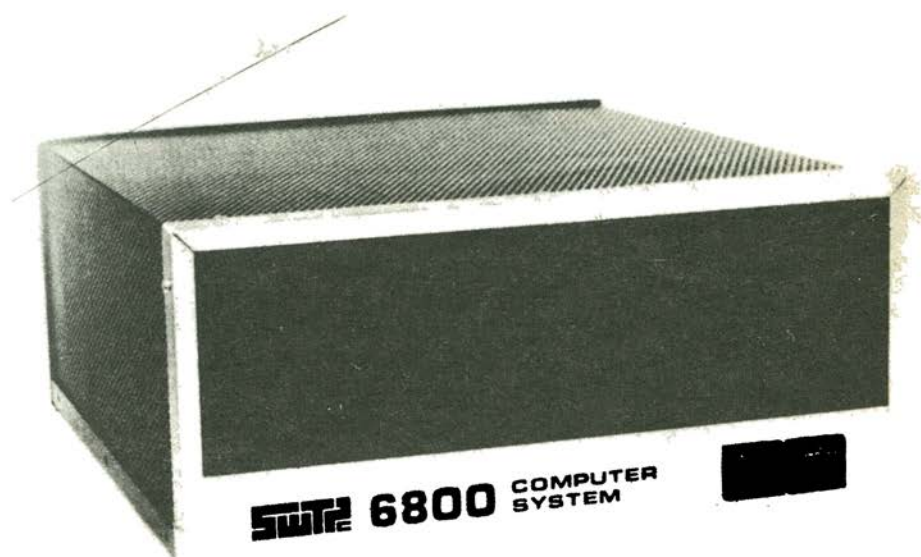



Fig. 10-26: Lo SWTP 6800.

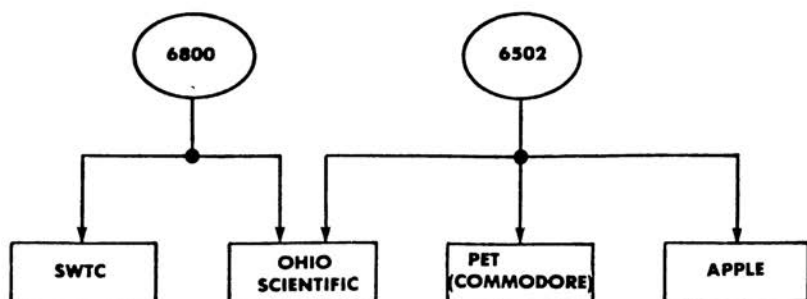



Fig. 10-27: L'albero genealogico dei microcomputers basati sul 6800 e sul 6502.

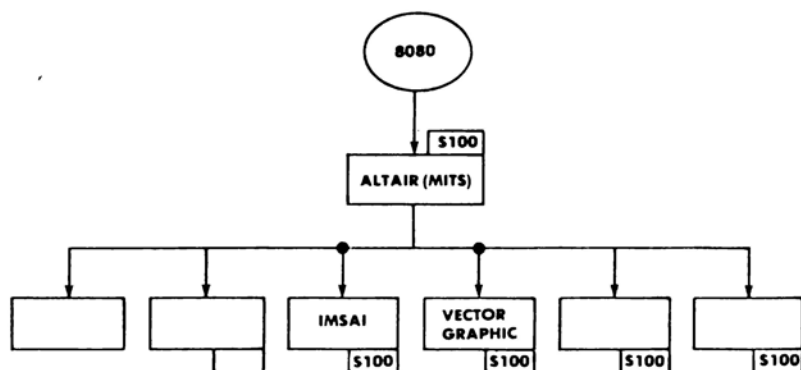



Fig. 10-28: L'albero genealogico dei microcomputers basati sull'8080.

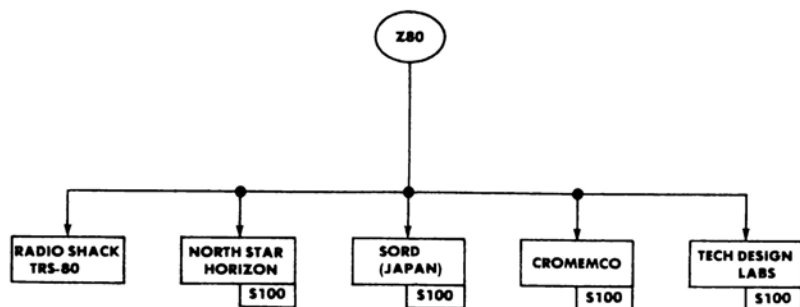



Fig. 10-29: L'albero genealogico dei microcomputers basati sullo Z-80.

PICCOLI SISTEMI COMMERCIALI

Un certo numero di costruttori oggigiorno fanno sistemi commerciali che possono usare un microcomputer o meno. Di seguito troverete una lista dei costruttori tradizionali di piccoli sistemi commerciali.

La IBM ha introdotto il 5110 (1978), un progetto brevettato, vicino come prezzo ai microcomputer, che offre BASIC ed APL; comprende una stampante, nastro e diskette. Esso ha l'usuale tastiera da macchina da scrivere.

SOMMARIO

I microcomputers sono stati classificati in tre categorie: a board singolo (didattica), sistema domestico (divertimento più processing limitato), e general purpose. Entro la categoria dei microcomputer general purpose, esiste un gran numero di prodotti che offrono varie combinazioni di caratteristiche all'interno del contenitore.

Per l'utente commerciale: sono adatti solo quei microcomputers che hanno:
- memoria completa (fino a 64K)
- BASIC completo risiedente nella ROM (deve risiedere nella ROM per efficienza)
- insieme completo di periferiche, o un bus standard al quale possano essere collegate altre periferiche
- software completo, compreso un DOS (Disk Operating System) ed un sistema di file
- disponibilità di packages per il commercio che possano essere eseguiti direttamente su questo processore

Per uso personale: la maggior parte dei sistemi offrono delle prestazioni simili e opzioni I/O differenti. L'analisi di cui sopra dovrebbe essere utile nel fornire dati di confronto.

CAPITOLO 11
ECONOMIA DI UN SISTEMA COMMERCIALE

IL COSTO REALE DI UN SISTEMA

Oggigiorno è possibile giocare con un sistema minimo con cifre che vanno da 150 a 500 dollari, compreso l'alimentatore e dispositivi minimi di input-output.
Se l'intenzione è di fare un po' di "seria programmazione", o persino di eseguire programmi commerciali, di solito sarà necessario:
1 - aggiungere della memoria
2 - aggiungere un terminale CRT
3 - aggiungere uno o più disk (se non sono compresi nel sistema)
4 - aggiungere una stampante (quasi mai compresa nel sistema)
Nel caso di un sistema commerciale, partendo dal prezzo del mobile del microcomputer di 1000 o 2000 dollari, il prezzo finale del sistema sarà forse quattro volte tanto il prezzo del mobile, o persino di più (aggiungete il prezzo del software).
Questo è sempre stato vero nei sistemi a computer. Per un sistema completo il costo principale è il costo delle periferiche.
Anche se il costo dei componenti elettronici diminuirà ancora, è improbabile che il costo delle periferiche diminuisca marcatamente, a meno che appaiano nuovi mercati di massa.

I costi nascosti

Al costo d'acquisto di un sistema devono essere aggiunti due principali fattori di costo addizionali: la manutenzione e la programmazione.
Nel caso di un sistema commerciale, una regola a lume di naso è di valutare la manutenzione l'1% del costo d'acquisto, per ogni mese.
La programmazione non ha un costo significativo nel caso di un sistema personale: è un programma di divertimento, o altri programmi che vengono acquistati a basso costo.

Nel caso di un sistema commerciale, i packages commerciali standard sono proprio tali: essi sono standardizzati e non si adatteranno mai esattamente alle necessità di ogni azienda, specialmente piccole aziende, che tendono ad essere altamente individualizzate.

È una pratica comune quella di schedulare almeno un programmatore a tempo ridotto come parte del costo permanente di un sistema a computer. Il programmatore avrà il compito di adattare i programmi standard, e di sviluppare packages specializzati richiesti dall'azienda.

Un terzo costo addizionale nascosto proviene dall'impatto di procedure computerizzate su di un'azienda. Nel caso ideale, il computer eliminerà la necessità di diversi posti di lavoro, o semplificherà marcatamente molti compiti.

In pratica, questo non è sempre il caso reale. In piccole ditte è difficile che il computer elimini qualche funzione. Il contabile sarà ancora necessario, come pure l'impiegato delle vendite. Tuttavia, il computer automatizzerà alcuni dei loro compiti, risparmiando del tempo, ma, cosa più importante, fornendo nuove risorse:

- resoconti e aggiornamenti istantanei.
- manipolazione dell'inventario
- rapporti e statistiche immediati

Una gran parte del tempo del personale di valore sarà libero per compiti più complessi, ed è probabile che tutti i lavori ripetitivi siano eseguiti più accuratamente: rapporti di vendita note di ordinazioni arretrate, riordinazione automatica.

In una situazione progredita, è facile che il microcomputer elimini la necessità di personale impiegatizio addizionale, consentendo un risparmio di mano d'opera.

Infine, la disponibilità di informazioni aggiornate ha pure un valore di management significativo.

In breve: in tutti i casi dove i criteri precedenti sono soddisfatti, cioè crescita o alto valore dell'informazione immediata, il computer darà come risultato risparmi immediati.

In altri casi, il suo uso può non essere necessario a meno che siano tenuti in considerazione altri servizi quali la gestione della mailing list.

Quando comprare

Poniamo che P sia il costo di acquisto del sistema.

Poniamo che M sia il costo di manutenzione.

Poniamo che A sia il costo addizionale (programmatore o altro).

Il costo del sistema è:

$$P + (M + A) \times T$$

dove T è il tempo (in mesi)

In pratica, si dovrebbe aggiungere L= perdita del periodo di un tempo che avviene quando il sistema viene installato, e quando viene fatta l'operazione d'acquisto.

Il costo totale è ora: $P + (M + A) \times T + L$.

Dalla parte positiva, si può assumere che il computer faccia risparmiare D dollari per mese riducendo il costo della mano d'opera ed eliminando servizi esterni di elaborazione dati (mailing list, libro paga).

Se D è piccolo, il sistema non "si pagherà da sé" almeno che uno prenda in considerazione il valore in dollari dei servizi addizionali che esso fornisce.

Se D è consistente, il sistema si sarà pagato da sé quando il costo del sistema sarà stato compensato dai risparmi, cioé quando

(1) $P + (M + A) \times T + L = D \times T$

cioè dopo un tempo $T = \dfrac{P + L}{D - M - A}$

Per farcela in 18 mesi, per esempio, il risparmio mensile D deve essere:

$$D = \frac{P + L + (M + A) \times 18}{18}$$

Infine, D dovrebbe includere non solo i risparmi di paghe ed i risparmi di EDP esterni ma dovrebbe tener conto delle entrate addizionali generate dalla disponibilità di:

- resoconti aggiornati
- riordinazioni immediate (= nessuna perdita di profitto dovuta a ritardo)
- marketing migliorato attraverso analisi e mailings (spedizioni)
- raccoglimento di ricevibili migliorato

La formula proposta in (1) è perciò probabile che sia il caso peggiore.

In aggiunta, il valore istruttivo del process risulterà in risparmi consistenti pochi anni dopo quando l'azienda sarà progredita.

Ci sono alternative?

Le alternative tradizionali al sistema in-house (in casa, in ditta) sono:

1 - servizio esterno di elaborazione dati
2 - affitto di un terminale time-sharing

Esaminiamo queste due alternative.

Elaborazione esterna di dati

Per compiti altamente specializzati, l'elaborazione esterna di dati può essere economica: libro paga, mailing list, tasse. Tuttavia ha degli svantaggi: costi crescenti con la crescita dell'azienda, tempo per andare in giro e per verificare l'affidabilità, sicurezza delle informazioni ed accessibilità incerte.

L'EDP esterno è tutt'ora una alternativa conveniente per molte ditte che trarrebbero solo benefici limitati da un computer in-house.

Infatti stanno sorgendo aziende di servizio, che usano i microcomputer per fornire questi servizi *specializzati* ad un gruppo di utenti.

Time-Sharing

È possibile noleggiare un terminale (CRT o stampante), e comunicare attraverso una linea telefonica con un computer centrale. L'utente paga il nolo del terminale, la linea, il tempo per il collegamento, ed il tempo di processing usato. Questa alternativa è molto comoda: uno è collegato ad un computer potente, equipaggiato con tutte le strutture. Tuttavia è un'alternativa costosa: il costo di noleggiare o acquistare un terminale modem (un dispositivo elettronico usato per trasmettere e ricevere dati sotto forma di un suono modulato in frequenza attraverso un sistema di comunicazioni) è una porzione consistente del costo di un microcomputer.

A meno che sia necessario del processing altamente specializzato (appare che l'accesso al computer sia gratuito) un sistema in-house oggigiorno è di solito più economico, a condizione che fornisca il servizio richiesto.

SOMMARIO

Sono state valutate ed esaminate le alternative in termini di benefici di costo di un sistema a microcomputer. Nella maggior parte delle aziende in sviluppo è probabile che venga raggiunta velocemente la soglia di redditività di un computer. Quindi purché il capitale sia disponibile, una computerizzazione a breve scadenza diventa un bene importante.

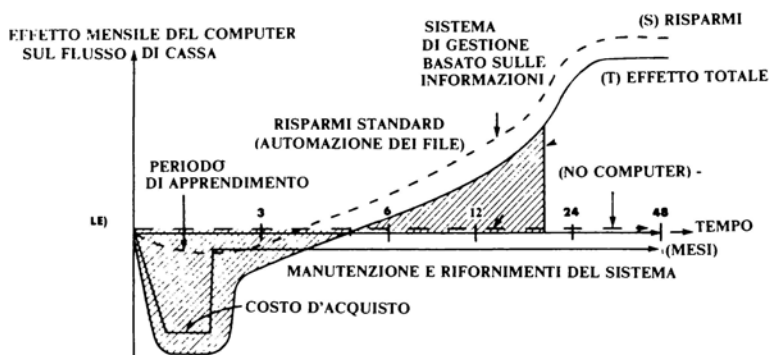
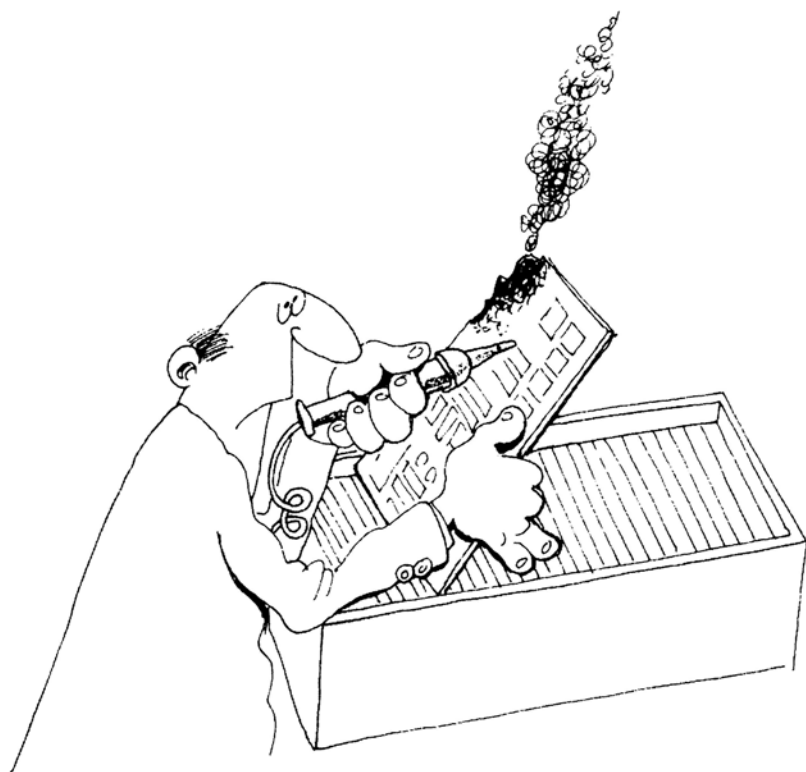



Fig. 11-1.

CAPITOLO 12

COME FALLIRE CON UN SISTEMA COMMERCIALE

INTRODUZIONE

Gli insuccessi qui saranno considerati "accidentali", cioè dovuti ad errore. Questo capitolo esplorerà le fonti di errore. Ogni utente di un sistema commerciale dovrebbe essere cosciente di queste trappole, così da evitarle.

Il filo conduttore per tutti i commenti che saranno presentati è che un sistema commerciale deve essere *sicuro ed affidabile*. I mal funzionamenti verranno ricondotti a tre fonti essenziali: hardware, software, procedure.

GUASTI HARDWARE

Si fa l'ipotesi qui che il sistema sia stato consegnato in buone condizioni di funzionamento. Le prime 100-200 ore di funzionamento sono chiamate periodo di *burn-in*. È durante questo periodo che è per lo più probabile che i componenti scadenti periscano. Molti costruttori e commercianti effettuano il "burn-in" di una unità prima di spedirla.

Consideriamo ora i guasti hardware che si verificano più facilmente nella vita dell'equipaggiamento.

Guasti meccanici

I guasti meccanici sono quelli che avvengono con maggiore probabilità, ed il candidato più probabile è la stampante. Per evitare tali guasti, in un ambiente produttivo viene di solito usata la manutenzione preventiva: vengono ispezionate le parti meccaniche, vengono pulite, ed aggiustate ogni x ore. Questa procedura individua la maggior parte delle avarie prima che accadano.

Guasti provocati dall'ambiente

Tutti i componenti di un sistema vengono definiti adatti al funzionamento entro un arco di temperatura e ad un livello di umidità specificati. Chiaramente, queste specifiche devono essere soddisfatte. Tuttavia, avarie dovute ad un momentaneo non soddisfacimento delle stesse sono di solito transitorie e non danneggeranno il sistema permanentemente.

Guasti all'elettronica

Come in qualsiasi sistema complesso, alcuni componenti elettronici potrebbero funzionare male. La raccomandazione generale in questo caso è di lasciare la diagnosi e la riparazione al costruttore. Questo implica la disponibilità di un servizio di riparazione efficiente in loco. Per tecniche dettagliate di individuazione di guasti il lettore con una mentalità tecnica è rimandato al nostro riferimento C207.

Sommario dei guasti hardware

Purché il sistema sia costruito seguendo le direttive accettate universalmente per un funzionamento affidabile, o in pratica se può essere ritenuto affidabile, i guasti hardware è probabile che siano minimi dopo i primi giorni. Essi di solito vengono affidati al servizio di riparazioni del costruttore o al fornitore locale. È di importanza critica per un sistema commerciale che sia disponibile un service in loco.

Tecniche speciali per aumentare l'affidabilità

La *parity* (parità) è stata usata a lungo come una delle tecniche favorite per verificare la corretta trasmissione a ritenzione delle informazioni. La parità consiste nell'aggiungere un bit extra ad ogni byte di dati per verificarne il contenuto. Una parità *pari* aggiungerà uno "zero" se il numero totale degli uno nel byte è pari, altrimenti aggiungerà un "uno". In altre parole, essa garantisce che il numero totale di bit sarà pari. Può anche essere usata una parità *dispari*.

Questa tecnica individuerà "errori di singolo bit" che sono quelli che avvengono con più probabilità: se un singolo bit cambia stato (da 0 a 1 o viceversa), il modulo di verifica della parità confronterà il bit di uguaglianza calcolato con quello immagazzinato insieme al byte, ed individuerà l'errore.

La parità è usata estesamente nei sistemi a media e grande scala. Non è quasi mai usata nei microprocessori.

Ci sono due ragioni:

1 - i sistemi a microprocessore sono molto più affidabili dei computer tradizionali, semplicemente perché essi usano molti componenti in meno (il tasso dei guasti aumenta rapidamente con il numero delle parti o delle interconnessioni).

2 - finora non c'è stata richiesta per affidabilità addizionale ad un costo e ad una complessità extra.

Una delle aree dove è probabile che si verifichi un'avaria è la memoria del sistema, principalmente perché essa usa un grande numero di componenti. Un'avaria temporanea durante un gioco di guerra spaziale non è deplorevole: semplicemente il gioco viene ricominciato. Un'avaria temporanea che spazza via ore di dati sugli acconti ricevibili è certamente deplorevole.

Poiché, in aggiunta, i sistemi commerciali è probabile che richiedano la più grande quantità di memoria RAM disponibile (40 o 48K byte), è più probabile che accadano malfunzionamenti della memoria. Ci saranno presto probabilmente sistemi di memoria per microprocessori equipaggiati con la opzione della parità, così da fornire un'affidabilità extra.

La parità viene usata a livello del byte. Tuttavia, nel caso di memorie di massa, come un disk o un nastro, non è possibile dedicare un bit extra a questa funzione. In questo caso, viene usato un intero byte (o diversi byte) alla fine di specifici block. Il byte contiene una "checksum" o un "CRC" (cyclic redundancy check) verifica ciclica sovrabbondante.

La *checksum* (verifica per somma) viene calcolata in base ad una semplice formula che coinvolge gli n byte precedenti (dove "n" dipende dal sistema). Se un byte viene cambiato, la checksum è pure cambiata, ed il modulo di verifica della checksum lo percepirà quando il blocco di dati viene letto. Il CRC usa una tecnica più complessa per calcolare i byte di CRC.

La checksum o il CRC vengono usati universalmente nel caso di disk o nastri. La checksum è un metodo semplificato, il CRC è un metodo più affidabile.

Infine un'altra buona pratica è la: *"read after write"* (lettura dopo scrittura): ogni volta che un blocco di dati viene scritto, dovrebbe essere riletto. Questo viene fatto talvolta nel caso di disks, e dovrebbe essere usato per files di importanza cruciale. È una funzione software. Di solito non viene attuata poiché essa rallenta l'operazione di scrittura. La sua disponibilità come opzione può essere di valore.

INCONVENIENTI DEL SOFTWARE

Ogni programma complesso dovrebbe essere considerato non corretto! Almeno in senso matematico: ci saranno quasi sempre alcune combinazioni di eventi che causeranno un malfunzionamento del programma. Tuttavia, la probabilità è piccola, e, si può sperare che ne risulti un danno lieve.

Poiché non c'è alcun modo per garantire che qualsiasi programma lungo, come qualsiasi sistema complesso costruito dall'uomo, sia totalmente senza errori, ogni sistema soffrirà occasionalmente dei difetti software.

Di solito i costruttori pubblicano aggiornamenti periodici di programmi in cui sono stati trovati difetti consistenti.

Per esempio, quasi ogni interprete BASIC, quando viene pubblicato per la prima volta, si comporta scorrettamente quando vengono usate delle particolari sequenze di istruzioni. Questi problemi vengono in seguito corretti.

Bisogna considerare che qualsiasi software, che non sia stato ancora collaudato da un gran numero di utenti, contiene dei difetti. Quindi, ogni nuova struttura software dovrebbe essere trattata con cautela in un ambiente commerciale.

Tuttavia questo aspetto non deve nemmeno essere esagerato: alcuni utenti possono non notare mai avarie importanti. Un sistema può non accettare un comando, e lo si dovrà semplicemente battere di nuovo sulla tastiera. Se viene usato un buon progetto per il sistema i malfunzionamenti sono spesso limitati alla necessità di ripetere un'operazione che non ha funzionato.

Persino il più grande sistema IBM occasionalmente subirà dei "crolli" catastrofici. Infatti, a causa della loro grande complessità, è più probabile che essi funzionino male, e che richiedano un certo numero di tecniche per aumentare l'affidabilità.

PROCEDURE

Bisogna distinguere diverse aree poiché questa è probabilmente l'area alla quale possono essere ricondotti i malfunzionamenti più gravi.

Poiché è difficile separare nettamente alcune procedure dalla loro realizzazione software, qui saranno anche discussi molti aspetti software.

Precisione dei dati

I dati dovrebbero essere verificati nel momento in cui vengono introdotti, e si dovrebbe anche mantenere la loro precisione mentre vengono elaborati dal computer.

Bisogna verificare la precisione dei dati, quando essi vengono introdotti. In molti casi questo può essere fatto tramite dei field check e limit checks.

Un field check verificherà, per esempio, che un "numero" non contenga accidentalmente dei caratteri, o altrimenti, due periodi.

Un limit check viene anche chiamato un "test di ragionevolezza". Per esempio, un mese dovrebbe essere tra l'1 ed il 12. Una storia d'orrore tradizionale sui computers è la seguente: una dattilografa inesperta (o stanca) in un ufficio pubblico introduce un ordine di 1.000.000 di dollari nel file degli acconti ricevibili, invece di 10.000 dollari.Il manager ha approvato un tasso di spesa del 30% delle somme ricevibili. Prima che l'errore sia individuato, potrebbero essere spesi fino a 300.000 dollari.

Un'altra storia d'orrore è "il caso di Mr. Smith". Viene ricevuto un pagamento di 780 dollari da Mr. Smith. Esso viene prontamente accreditato sul suo conto, che mostrava un bilancio di 180 dollari di debito. Viene emesso un assegno di rimborso di 600 dollari. Un mese più tardi un certo Mr. Smith adirato telefona, e chiede perché lui ha ricevuto un "ultimatum" computerizzato, quando lui ha pagato completamente. Voi avete indovinato il perché, c'erano due "Mr. Smith" nel file. Il pagamento fu accreditato a quello sbagliato.

Questa situazione dovrebbe essere verificata o da un uomo, o anche da una procedura software. Idealmente, il programma segnalerà la situazione degli "Smith multipli".

Rinforzare i controlli

È un principio commerciale universale quello che i compiti debbano essere separati: la persona che verifica i dati non deve essere quella che li introduce.

Questo riduce il rischio di errori accidentali, ed introduce anche un controllo contro modifiche dolose, o altre modifiche.

Anche da un punto di vista procedurale, è una pratica commerciale vitale quella di mantenere una *audit trail* (una pista di audit) entro tutti i file: semplicemente tutti i file o transazioni devono includere riferimenti appropriati ai file degli originali o ai documenti che permettono una verifica.

Per esempio un pagamento deve essere correlato come minimo alla data, numero di assegno, e numero della bolletta. In modo simile, una lista di commissioni deve essere correlata alle transazioni da cui la lista fu originata.

In altri termini bisogna elencare sufficienti cross-references (riferimenti incrociati) per permettere una verifica completa dei dati.

Controllare i digit

Nelle situazioni in cui vengono usati codici lunghi come negli inventari, è essenziale usare redundant encoding, (codificazione sovrabbondante), o i digit di verifica.

Il *redundant econding* userà codici come:

TUBE-204211
BOLT-418182
ASBL-881921
FRME-329137

Le lettere al principio del codice identificano il prodotto per un operatore umano. Ogni "tubo" ha un codice "204". Ogni "bullone" ha un codice "418". Il programma verificherà che i codici corrispondano alle lettere. Accadranno errori tutt'al più entro i vari tipi di tubi o bulloni disponibili, cioé entro gli ultimi 3 digit.

Questo metodo è facile da realizzare, ma spreca dello spazio.

Un digit di verifica è un digit extra aggiunto al codice del quale rivelerà un errore di trasposizione, o un errore di un singolo digit.

Per esempio: 881921-5
5 è il digit di verifica. È calcolato moltiplicando "1" per "1", poi "2" per n^2, poi "9" per n^3, ecc. addizionando poi i 6 numeri. La somma é divisa per n. Se il resto é R, il digit di verifica é n-R.La scelta di n varia. Questo metodo é quasi sempre usato nel verificare i numeri degli acconti, poiché l'errore più frequente è la trasposizione, cioé scrivere 21224138 invece di 21221438.

Sicurezza dei dati

I file dovrebbero essere sicuri sia nel caso del mal funzionamento della macchina sia nel caso di errori o interferenze dell'uomo.

Naturalmente, tutti i file importanti devono essere duplicati ad intervalli regolari, e protetti.

In maniera ugualmente importante dovrebbero essere disponibili protezioni software, come parole d'ordine o altri meccanismi di protezione dell'accesso, per un buon sistema di file.

Si può imporre una *parola d'ordine* per accedere ad un file specifico come Read or Write (Lettura o Scrittura). Non deve essere echoed (fare l'eco sul CRT). Non deve essere facilmente accessibile all'interno del sistema. Una volta che è stato creato, un file deve essere protetto da accesso non autorizzato o modificazioni da parte della persona sbagliata.

Possibilità di *protezione dei files* associate ad un buon sistema di file sono l'abilità di stabilire "attributi di accesso" associati ad un file, come "la protezione della scrittura".

Per finire, ciò vuol dire che non si dovrebbe sottovalutare qualcosa come una chiave per intervenire sul sistema.

Lo Shock da computer

Anche gli effetti di computerizzazione possono essere gravi. Di solito, a meno che il personale operativo sia stato addestrato per la transazione (da non computerizzato a computerizzato), potrebbero verificarsi inizialmente dei ritardi operativi, che avrebbero come risultato un rendimento degradato per un certo periodo di tempo. Tuttavia, c'è di solito una transazione veloce ad un'elevata efficienza, purché vengano usate le procedure corrette.

SOMMARIO

Ci si deve aspettare mal funzionamenti software ed hardware limitati. Essi non dovrebbero influenzare consistentemente il funzionamento del sistema purché vengano prese le precauzioni che sono state indicate. Il rischio maggiore sta di solito nelle procedure (sia di programma che di gestione) usate sul sistema. Il buon senso, intuito commerciale e il consiglio da parte di un utente esperto sono le risorse basilari necessarie.

ACQUISTO DEL SISTEMA - UN SOMMARIO

1 - Ce ne deve essere la necessità

La natura e la quantità del lavoro devono giustificare il cambiamento. Questo può essere misurato dal numero di transazioni o resoconti identici che devono essere fatti. Inoltre il sistema a computer può essere giustificato dai singoli problemi che può risolvere in situazioni particolari, coma la gestione dell'inventario, mailing list, resoconti di management.

2 - **Quando comprare?**

I prezzi per i componenti elettronici continuano a diminuire ogni anno, mentre i prezzi delle periferiche tendono ad essere stabili o a diminuire lentamente. Il sistema di domani sarà sempre più economico di quello di oggi. Tuttavia, un sistema a computer significa risparmiare N dollari per mese. Posticipare la sua installazione di un mese è equivalente ad una perdita di m x N dollari.

3 - **Quale comprare?**

Sono state presentate tutte le principali alternative ed apparati disponibili.

Altre precauzioni standard

- Creare un back-up (supporto) su un disk, nastro o carta, alla fine di ogni giornata, così che qualsiasi file danneggiato o perduto possa essere ricreato.
- Proteggetevi dalle fluttuazioni della rete di alimentazione se questo problema affligge la vostra area.

CAPITOLO 13

AIUTO

OTTENERE INFORMAZIONI

Esistono molte fonti di informazioni che faciliteranno la decisione:
- pubblicazioni
- clubs
- grandi magazzini di computer
- consulenti
- istituzioni didattiche o aziende

Saranno ora riesaminate le nuove fonti principali

RIVISTE

La maggior parte delle riviste tradizionali sull'elettronica e sui computers presentano regolarmente articoli sui microcomputers. Tuttavia, sono sorte numerose nuove pubblicazioni dal 1976 che sono destinate specificatamente agli utenti o ai progettisti di microcomputers. Esse sono elencate alla fine di questo capitolo. È difficile assegnarle ad un campo specifico poiché esse si evolvono rapidamente.

Tuttavia fra le riviste americane, il Dr. Dobb's Journal and Creative Computing contiene sempre una consistente sezione software, mentre le riviste Interface Age, Byte e Kilobaud contengono anche progetti hardware.

La maggior parte delle riviste è disponibile nelle edicole internazionali.

In Italia esiste una rivista dal nome Bit, molto apprezzata in questo settore.

CLUB E GRANDI MAGAZZINI DI COMPUTER

Sono noti un certo numero di club per hobbisti ed i loro indirizzi vengono di solito pubblicati nelle riviste di cui sopra, in particolare nelle pubblicazioni della People's Computer Company.

I grandi magazzini di computer sono un fenomeno di vendita al dettaglio. Probabilmente il primo fu creato a Santa Monica da Dick Heiser nell'estate del 1975: la Arrowhead Computer Company. Poco dopo il primo Byte Shop fu installato a Mountain View (California) da Paul Terren ed oggi è diventato una delle più grandi catene di grandi magazzini di computer.

Sono stati aperti centinaia di magazzini, in tutte le città principali negli USA ed in Europa, che vendono esclusivamente sistemi e prodotti dei microcomputer sia agli hobbisti che agli uomini d'affari. I magazzini di computer possono offrire servizi di valore: mostra di sistemi, manutenzione di riparazione, packages speciali (sia hardware che software), libri e riviste, consigli competenti, lezioni, tabelle.

È opinione comune che la vendita di computers general purpose richieda una capacità non disponibile generalmente in altri magazzini al dettaglio, così che molti di questi punti di vendita specializzati stanno prosperando. Persino compagnie come la IBM e la Tandy stanno aprendo magazzini di computer specializzati.
Un buon magazzino di computer è uno dei vantaggi principali per un acquisto comparativo, e per l'assistenza in loco quando si inizia.

Tuttavia, naturalmente non è assicurata la sopravvivenza a lungo termine di molte di queste avventure.

CONSULENTI

I consulenti sono stati una risorsa tradizionale a cui fare appello per l'installazione di computer medio/grande. Il *libero* consulente del giorno d'oggi è il competente possessore o agente di un magazzino di computer.

I consulenti tradizionali sono semplicemente troppo costosi proporzionalmente al costo di sistemi microcomputer.

ADDESTRAMENTO

L'addestramento è sempre l'investimento migliore prima o dopo l'acquisto. Sono disponibili una gran varietà di mezzi. Lezioni dal vivo vengono spesso condotte in magazzini di computer o nelle università, o da compagnie commerciali rispettabili. Sono disponibili video-programmi per gruppi. Infine libri, o corsi autodidatti sono quasi imperativi per una scelta e comprensione efficienti.

MOSTRE DI COMPUTERS

La mostra che avviò il tutto fu probabilmente la Personal Computing Show organizzata ad Atlantic City nell'Agosto 1976 da John Dilks.

Organizzata nella tradizione dei convegni dei radio hobbisti, ci si aspettava che attirasse un piccolo numero di visitatori. Ne vennero migliaia, e gli osservatori industriali capirono che era in atto una nuova tendenza; il personal computing era una realtà.

Il successo di questa mostra diede il via ad un'ondata di mostre simili in tutti gli USA. Una delle più riuscite sulla West Coast e stata la West Coast Computer Faire, organizzata da Jim Warren (15.000 partecipanti nel 1978).

Le mostre di computers offrono un'opportunità unica nel loro genere per osservare quasi tutti gli apparati disponibili in una volta ed ottenere comodamente informazioni e documentazioni. Esse sono opportunità educative di alto valore, e vengono pubblicati regolarmente i calendari nelle riviste di microcomputers.

ALL'ESTERO

Un piccolo numero di negozi di microcomputers e riviste specializzate esistono ora in tutto il mondo. In Europa un riferimento utile é la Euromicro, The European Association for Microprocessing (indirizzo elencato in altre pubblicazioni). Anche nelle nazioni più importanti vengono organizzate mostre di computers, in scala spesso comparabile a quella degli U.S.A. Nel 79 è stata organizzata a Milano la prima mostra italiana del genere. La mostra si chiamava Bit ed era organizzata dall'omonima rivista del Centro Commerciale Americano. Il successo della manifestazione è stato notevole a conferma dell'interesse che questi sistemi hanno riscosso anche in Italia.

RIVISTE SUI MICROCOMPUTERS

BIT
Jackson Italiana Editrice
Piazzale Massari, 22
20125 Milano

BYTE
70 Main Street
Peterborough
New Hampshire 03458

CALCULATORS and COMPUTERS
DYMAX, P.O. Box 310
Menlo Park
CA. 94025

CREATIVE COMPUTING
P.O. Box 789-M
Morristown, NJ

DR. DOBBS JOURNAL
Box 310
Menlo Park
CA. 94025

EUROMICRO JOURNAL
313 rue Lecourbe
75015 Paris
France

INTERFACE AGE
Box 1234
CA. 90701

KILOBAUD
1001001, INC.
Peterborough
New Hampshire 03458

PEOPLE'S COMPUTERS
1263 El Camino Real, Box 5B
Menlo Park
CA. 94025

PERSONAL COMPUTING
401 Louisiana S.E.
Albuquerque
New Mexico 87108

SCCS INTERFACE
P.O. Box 5429
Santa Monica
CA. 90405

CAPITOLO 14

DOMANI

SOMMARIO

In tutto questo volume sono stati introdotti tutti i concetti principali per quel che riguarda i microcomputers. Informazioni tecniche addizionali si possono trovare nella sezione delle appendici.

Il funzionamento di un sistema completo dovrebbe essere chiaro, come pure le alternative dettagliate per la scelta delle apparecchiature.

Il lettore dovrebbe aver acquisito tutta la conoscenza necessaria per una scelta razionale. Il prossimo passo può essere quello di imparare di più e vari volumi simili estendono la conoscenza nella direzione hardware (C201, C207), o nella direzione software (C202).

Tuttavia il prossimo passo potrebbe essere l'acquisto di un sistema per fare un po' di pratica (e divertimento). L'utilizzazione del computer è stata qualificata come il gioco estremo, poiché estende le funzioni dell'intelligenza di una persona, e la maggior parte degli utenti ne vengono affascinati rapidamente.

Ma questo campo si sta muovendo rapidamente, cosa accadrà?

DOMANI

Prezzo e miniaturizzazione

Col costo dei componenti elettronici che continua a diminuire e con la miniaturizzazione in continuo aumento, è probabile che la maggior parte dei componenti meccanici ed elettronici dei sistemi odierni diventeranno eventualmente elettronici, con una corrispondente diminuzione del prezzo e del volume: tastiere elettroniche, display elettronici, memorie di massa elettroniche. Sono già disponibili microcomputer a singolo chip che hanno diverse migliaia di bytes di memoria nel chip. Una volta che alcuni boards saranno stati standardizzati, essi diventeranno candidati per la realizzazione su singolo chip.

Un sistema completo, con le funzioni attuali, potrebbe probabilmente essere rea-

lizzato nel volume di un calcolatore tascabile, ad un prezzo simile. Il problema principale resta un input-output a dimensione d'uomo, e probabilmente resterà il fattore di costo maggiore.

La funzione di computing in sé è diventata essenzialmente libera, così che può ora essere introdotto un numero illimitato di nuove applicazioni. Ne svilupperete anche voi una?

Conclusione

Chi avrebbe previsto nel 1975 la situazione del settore odierno? Sebbene esistessero molte premesse, la realtà ha superato la "seconda rivoluzione industriale". Nessuno può prevedere nemmeno a breve termine il futuro a causa della crescita esplosiva di questa nuova industria.

Anche se non tutti useranno i microcomputers, la vita di tutti verrà cambiata dal loro uso. Una delle vostre risorse di maggior valore domani sarà la stessa di oggi: la conoscenza. Si spera che questo libro sia stato un passo su questa via.

APPENDICE A

LOGICA DEL COMPUTER

INTRODUZIONE

Questa è una breve introduzione ai concetti ed ai simboli base usati per i circuiti digitali nei sistemi microcomputer.

Tutte le informazioni sono presentate in formato binario, e processate, modificate o trasmesse da questi circuiti.

I CIRCUITI LOGICI BASE

I circuiti logici base sono l'AND, OR, NOR con alcune variazioni come lo XOR, NAND, NOT. Essi sono usati in combinazioni per fornire tutte le altre funzioni.

AND

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

XOR

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

NAND

A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

NOR

A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

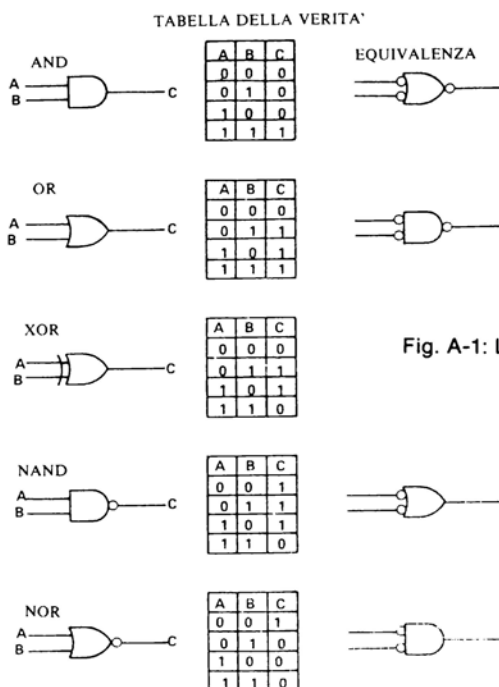



Fig. A-1: Le funzioni logiche di base.

La funzione di ognuno di questi "gate" viene rappresentata da una tabella della verità (tavola che esprime il valore dell'uscita in corrispondenza del valore degli ingressi - vedi fig. A-1).

Per esempio, l'uscita "C" di una porta AND sarà "1" solo se sia "A" che "B" sono "1". Altrimenti sarà "0".

Similmente, l'uscita "C" di una porta OR sarà "1" se uno dei due ingressi o entrambi sono "1".

I Flip Flops

Abbiamo anche bisogno di dispositivi di memoria che possano memorizzare uno stato ("0" o "1") e possono cambiare stato quando lo si comanda.

Nella fig. A-2 e A-3 appaiono i vari tipi di flip-flop. Un "flip-flop D" introduce un ritardo. Gli ingressi sono sempre complementari.

- Un "flip-flop J-K" modula la sua risposta in funzione della combinazione degli ingressi (vedi la tabella della verità). L'uscita Q è "clocked", e cambia solo dopo l'impulso di clock.
- Un "flip-flop - RS" é un flip-flop Set-Reset, il più semplice. E' un *latch*, ed é in posizione *set*, quando Q = 1, asincrono o sincrono (se viene usato un clock).

In fig. A-4 appaiono le possibili realizzazioni dei flip-flop mediante porte elementari.

Un *registro* è un insieme di flip-flop (8 per un byte).

Infine nelle fig. A-5 e A-6 appare una varietà di simboli elettronici ed elettrici.

J
CLOCK
K
Q
Q

Fig. A-2: Un flip-flop JK.

t_n	t_{n+}
D	Q
0	1
1	1

t_n		t_{n+1}
J	K	Q
0	0	Q_n
0	1	0
1	0	1
1	1	Q_n

t_n		t_{n+1}
S	R	Q
0	0	Q_n
0	1	0
1	0	1
1	1	X

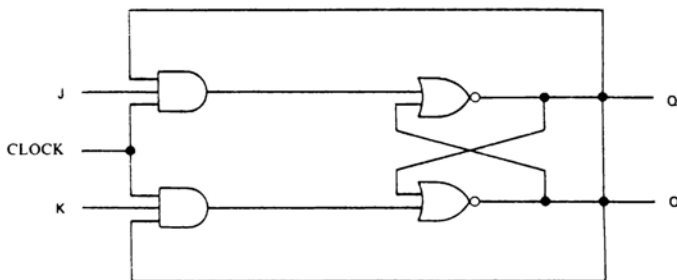
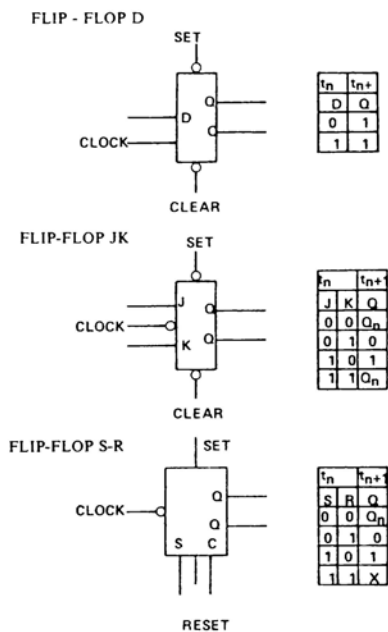
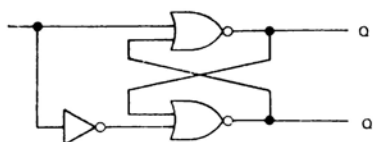



Fig. A-3: Flip-flop.

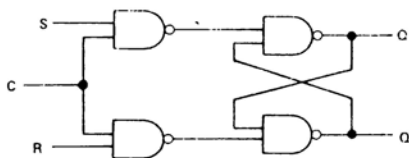
IL FLIP-FLOP D HA UN INVERTER EXTRA

IL FLIP-FLOP S-R CLOCKED USA DUE PORTE NAND

Fig. A-4: Flip-flop con i gates.

TERRA VERA

"TERRA" DI RIFERIMENTO

COLLEGAMENTO

NON COLLEGAMENTO

CRISTALLO

LUCE

RESISTENZA

REOSTATO (RESISTORE VARIABILE)

BOBINA

DIODO

LED

DIODO ZENER

TRANSISTOR NPN

TRANSISTOR PNP

INTERRUTTORE SPST

COMMUTATORE SPDT

PULSANTE

TRASFORMATORE

BATTERIA

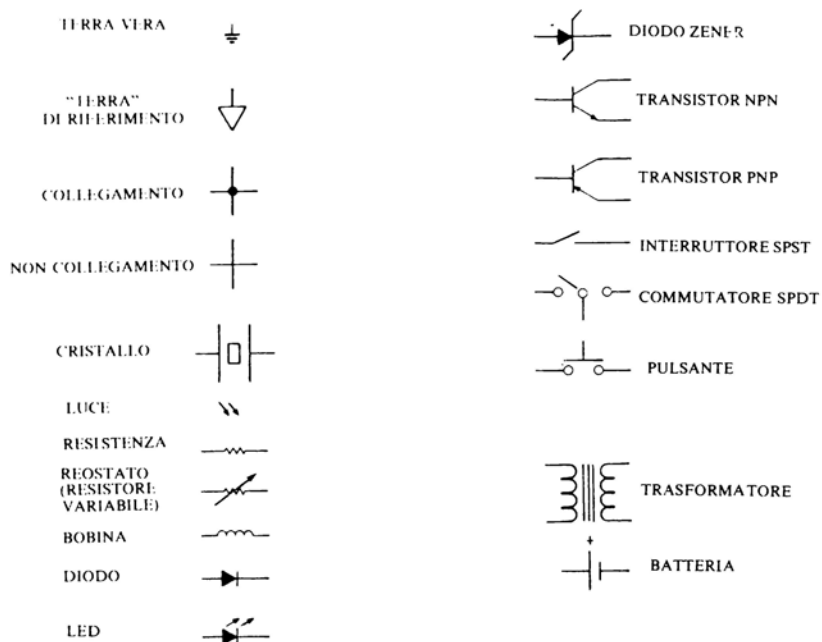



Fig. A-5: Simboli.

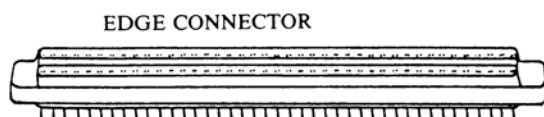



Fig. A-6: Un connettore edge.

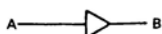
A	B
0	0
1	1

A	B
0	1
1	0

Fig. A-7: Amplificatori.

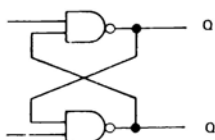



Fig. A-8: Il latch.

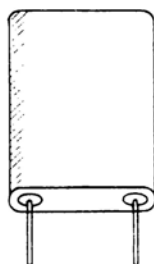
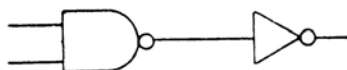
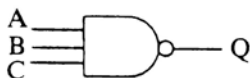



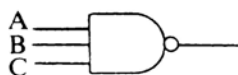
Fig. A-9: Il cristallo (quarzo).

ESERCIZI

- Qual'è il nome della porta realizzata dal circuito seguente?

Risposta: AND

- Costruite la tabella della verità per una porta AND a 3 ingressi:

Risposta:

A	B	C	Q
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

• Costruite la tabella della verità per una porta NAND a 3 ingressi:

Risposta:

A	B	C	Q
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

APPENDICE B

BITS E BYTES

IL SISTEMA BINARIO

Questo capitolo è una breve introduzione al sistema binario, usato all'interno del computer per rappresentare le informazioni. Generalmente, i numeri sono semplici da rappresentare. Guardiamo alcuni esempi:

3 è rappresentato da 11
5 è rappresentato da 101

La rappresentazione binaria usa gli 0 e gli 1 per rappresentare tutti i numeri. L'equivalente decimale è calcolato come segue:

11 è $1\times2+1\times1 = 3$
101 è $1\times4+0\times2+1\times1 = 5$

Il bit (binary digit) più a destra rappresenta $2^0=1$. È chiamato il "bit meno significativo" (LSB less significant bit). Spostandosi da destra a sinistra, come nel sistema decimale, ogni bit successivo rappresenta una corrispondente potenza di 2. Così, in "101", il bit più a destra rappresenta 1, il seguente $2^1=2$, ed il seguente $2^2=4$.

Noi sappiamo come ottenere l'equivalente decimale: se $b_n b_{n-1} \ldots b_1 b_0$ è il numero binario , il suo equivalente è:

$b_n \times 2^n + B_{n-1} \times 2^{n-1} + \ldots + b_1 \times 2 + b_0$

dove le b sono degli 0 e degli 1.

DA DECIMALE A BINARIO

La conversione inversa è altrettanto semplice:

16 diviso per 2 è 8, il resto è **0**
8 diviso per 2 è 4, il resto è **0**
4 diviso per 2 è 2, il resto è **0**
2 diviso per 2 è 1, il resto è **0**

l'ultimo quoziente, più i resti sono, dal basso verso l'alto: 10000. Questo è "16". Un altro esempio:

11 diviso per 2 è 5, il resto è **1**
5 diviso 2 è 2, il resto è **1**

. 2 diviso per 2 è 1, il resto è **0**
La risposta è 1011.

Esercizio: *convertire 10000 e 1011 in decimale così da verificare le conversioni precedenti.*

RAPPRESENTAZIONI NUMERICHE

Ora abbiamo mostrato che i numeri decimali possono essere rappresentati da bit. In pratica, i microcomputers attuali strutturano tutte le informazioni in gruppi di 8 bit, o byte.

Esercizio: *Qual'è il numero intero più grande che un byte possa rappresentare?*

Risposta: $2^8 = 256$

Esercizio: *Qual'è il numero intero più grande che due bit possano rappresentare?*

Risposta: $2^{16} = 65.536 = 64K$

Aritmetica binaria

Le operazioni aritmetiche sono eseguite in maniera simile al sistema decimale:

$$\begin{array}{rl} 0101 & (5) \\ +\ 0110 & (6) \\ \hline =\ 1011 & (11) \end{array}$$

Le regole dell'addizione sono rappresentate da:

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 10$ (o "0" col riporto di "1")

Numeri negativi e frazionari

Il prossimo problema è quello di rappresentare i numeri negativi ed i numeri frazionari. Ma questo è il soggetto di un altro libro (vedi ref. C201 o C202).

Brevemente i numeri provvisti di segno vengono generalmente rappresentati con la "notazione complessa del 2", mentre i numeri frazionari richiedono una rappresentazione a virgola mobile.

CARATTERI

I caratteri sono (quasi) universalmente codificati nel codice ASCII a 8 bit. La tabella di conversione ASCII appare a pag. 62.

Sette bit possono codificare $2^7 = 128$ caratteri differenti. Questo è sufficiente per:
26 lettere dell'alfabeto (da A a Z), maiuscole
26 lettere dell'alfabeto (da a a z), minuscole
10 cifre decimali (da 0 a 9)
66 simboli speciali.

Non sono quasi mai richiesti più caratteri, così che questo codice viene usato universalmente. In pratica, vari "caratteri" sono caratteri di *controllo*, che non vengono stampati, e che sono usati come comandi o informazioni di status tra il computer ed il terminale. Per esempio, ESC (ape) è usato generalmente come comando di "STOP THIS" (interrompi questo).

Domanda: *Per che cosa viene usato l'ottavo bit?*
Risposta: *È il bit di parity (parità)*

Illustriamo ora l'uso del sistema binario in un esempio finale:
L'utente batte sulla tastiera "THIS IS 12".
Il programma monitor o il programma editor prenderanno questi caratteri dalla tastiera, e li immagazzineranno nella memoria in forma binaria.

Assumiamo che il codice per:

SPACEè 10100000
T é 11010100
H è 01001000
I è 11001001
S è 01010011
1 è 10110001
2 è 10110010

La rappresentazione della string (fila) "THIS IS 12" è:

11010100
01001000
11001001
01010011
11001001
01010011
10100001
10110010

Domanda: *Non c'è un errore nella rappresentazione di 12?*

Risposta: no. Quando le cifre appaiono in uno "string", esse sono trattate come caratteri ASCII, e il "12" è rappresentato dal carattere ASCII per l'"1", seguito dal carattere ASCII per il "2". Un programma Editor non conosce i numeri. Esso manipola solo caratteri. Se viene usato un "assembler", esso dovrà "sapere" che un numero è un numero, di solito trovandolo alla destra di un operatore, come =, + o

Esso allora codificherà il numero in binario (di solito complemento a due), così che si possa eseguire facilmente su di esso l'aritmetica binaria.

Domanda per il lettore minuzioso: chi esegue la conversione della chiusura del tasto sulla tastiera, al byte che rappresenta il carattere: 1 - il microprocessore - 2 - il codificatore della tastiera.

Risposta: se la tastiera ha un codificatore (le tastiere alfanumeriche di solito ce l'hanno), è il codificatore che compie l'azione. Altrimenti (piccole tastiere a 16 o 24 tasti ne sono di solito "prive"), la codificazione viene fatta da un programma di conversione (esso è una parte del programma "monitor").

APPENDICE C

SISTEMI DI TRASMISSIONE DI BASE DEL COMPUTER

PARALLELO E SERIALE

Esistono quattro opzioni principali per trasmettere le informazioni tra un computer e le sue periferiche: parallelo oppure seriale, sincrono oppure asincrono.

Parallelo:

La trasmissione parallela nel caso di microcomputer standard coinvolge 8 bit, cioé un bus a 8 fili. Naturalmente, la trasmissione parallela è più veloce della trasmissione seriale, nella quale i bit vengono trasmessi solo uno dopo l'altro.

Ovunque sia richiesta velocità, bisogna usare la trasmissione parallela.

Questo è sempre ciò che avviene all'*interno* della scatola del microcomputer, dove i board devono scambiare informazioni alla più alta velocità possibile. Quindi il bus che opera sul *motherboard* (board principale) e che trasporta i segnali tra i board è sempre parallelo. Questo è il caso, per esempio, del bus S-100, già descritto.

All'esterno del microcomputer, un bus parallelo richiede 8 linee per i dati, più molte linee di controllo, per segnali di sincronizzazione. Sarà necessario per esempio per una stampante parallela (veloce).

Seriale:

La trasmissione seriale consiste nell'inviare bit su una singola linea uno dopo l'altro. Per distinguere due bit successivi, è necessario anche un segnale di "clock". Il vantaggio ovvio è il basso costo della linea di comunicazione (appena due fili). Tuttavia, a causa della limitata larghezza di banda (in termini di frequenza massima) disponibile per una trasmissione affidabile, questa tecnica limita la velocità di trasmissione. Essa richiede pure un buffer a 8 bit ad ogni estremo per accumulare un carattere.

Tuttavia è semplice ed economica, ed è usata per la maggior parte delle "periferiche lente", come telescriventi e terminali CRT.

La velocità utilizzabile varia da 110 baud a 9600 baud. Nel mondo digitale, i baud sono il numero di bit per secondo, o bps.

Una Teletype è un terminale a 110 baud (10 caratteri per secondo, che usano 11 bit ciascuno).

Una Decwriter LA 36 è un terminale a 300 baud.
Un CRT standard funziona fino a 9600 baud.

Interfaccia seriale:
Le due interfacce seriali standard più usate sono l'RS232C ed il 20mA current loop. Il 20 mA loop è l'interfaccia usuale per la Teletype, in cui "Mark" è 1 e "space" è 0.
L'RS-232C viene usato dai CRT e per connessioni telefoniche.
Da +3 a +15Volt è usato per lo "0", e da −3 a −15V per l'"1".

SINCRONO E ASINCRONO

Le informazioni possono essere inviate un bit alla volta: questa è una tecnica asincrona. Il terminale, "non sa mai" quando arriva un carattere. Per riconoscere un carattere, devono viaggiare alcune informazioni di sincronizzazione insieme al carattere.
Così, la Teletype usa un codice ASCII a 8 bit per il carattere, preceduto da un bit di "START", e concluso da due bit di "STOP".
Questa è la ragione per cui una Teletype a 10 cps (caratteri per secondo) richiede 110 baud: ogni carattere richiede 11 bit.

La comunicazione asincrona viene usata ovunque sia possibile, poiché essa è semplice ed affidabile.

La trasmissione sincrono invia "pacchetti" di dati o blocchi di dati che devono essere sincronizzati accuratamente. Essa offre la potenza dell'alta velocità, ma richiede molta logica in più ed è, perciò, più costosa. Può essere usata per comunicazioni ad alta velocità, con un altro computer per esempio.

AL DI LA' DEL SISTEMA MICROCALCOLATORE

Un microcomputer può comunicare con periferiche distanti (o altri microcomputer) attraverso le linee telefoniche (o, per brevi distanze con un twisted pair - tipo di cavo in cui i due conduttori isolati sono avvolti uno intorno all'altro). Per trasmettere affidabilmente informazioni a lunghe distanze sulle linee telefoniche, esse devono essere codificate in frequenze audio (in forma seriale!). Questo viene fatto da un *modem*.
Un modem è un modulatore-demodulatore. Esso codifica i dati binari seriali in frequenze, e decodifica frequenze in bit seriali. La tecnica usata più di frequente è la FSK, o "Frequency-Shift-Keying": una certa frequenza è la "portante", e dà l'in-

formazione di "nessun dato". Un "1" viene trasmesso inviando una frequenza più alta ("shifting" it up = facendola scorrere verso l'alto). Uno "0" è trasmesso inviando una frequenza più bassa.

Una frequenza può essere usata per trasmettere, ed un'altra, ben distinta dalla prima per ricevere. Una trasmissione in cui i dati possono essere inviati in entrambe le direzioni viene detta "full duplex". Altrimenti è "half duplex".

Per alte velocità di trasmissione, possono essere noleggiate linee telefoniche di qualità superiore. Questo è il modo in cui i computer più grossi vengono allacciati alle reti.

SOMMARIO

Entro la scatola del microcomputer, tutti i trasferimenti di informazioni sono paralleli.

All'esterno di essa, i trasferimenti a terminali lenti sono seriali, tramite un connettore RS-232C, o uno 20 mA current loop; ed i trasferimenti a dispositivi ad alta velocità sono paralleli, tramite connettori e port specializzati.

Inoltre il microcomputer può anche trasmettere informazioni tramite linee telefoniche, usando un modem.

APPENDICE D

FILES E RECORDS

PERCHÈ?

Un'unità di informazioni per gli scopi dell'utente viene chiamata file. Un file può essere un programma, una mailing list, o una lista di acconti ricevibili.

All'utente conviene assegnare un nome ad ogni file che usa. Per esempio dirà:

"LOAD (FROM THE DISK) PAYROLL"

(Carica "dal disk" libro paga).

Tuttavia, i files possono essere molto differenti, e pongono molti problemi pratici:

- un file può avere qualsiasi lunghezza
- un file può crescere o rimpicciolirsi
- può essere che si debba accedere in maniera efficiente alle informazioni al centro del file
- devono essere disponibili informazioni circa il file: quant'è lungo? È binario o ASCII?
- il suo contenuto deve essere corretto e sicuro

Per risolvere questi problemi, sono state sviluppate delle tecniche che hanno dato come risultato una varietà di opzioni di progetto per "strutturare i file". Rivediamo le principali:

ASSEGNAZIONE DELLA MEMORIA

Ogni file può variare nelle dimensioni. Un file di "libro paga", per esempio, di solito cresce, e talvolta si rimpicciolisce.
Tuttavia ogni mezzo d'immagazzinamento, sia disk che nastro, è finito. Il modo più semplice è di assegnare ad un file un intero diskette o un intero nastro: esso avrà parecchio spazio per crescere.

Sfortunatamente in ogni momento sono collegati al sistema solo un (o due) nastro o diskette. Una simile soluzione "spreca spazio", cioé restringe il numero di files a cui l'utente potrà accedere simultaneamente (c'è solo un file per ogni dispositivo d'immagazzinamento).

Un miglioramento ovvio è quello di definire una lunghezza massima per il file, diciamo mezza cassetta. In questo modo ci saranno due file per cassetta.

Se inizialmente il file è molto piccolo, è probabile che questa soluzione sprechi lo stesso i 3/4 dello spazio disponibile.

Files sequenziali

Dal punto di vista di uno sfruttamento dello spazio, la cosa migliore sarebbe quella di immagazzinare i files "*sequenzialmente*", cioé uno dopo l'altro.

Sfortunatamente, se un file cresce, allora tutti gli altri devono essere spostati, o altrimenti bisogna rimuovere quello subito dopo. Questo è un processo lungo e oneroso.

Questo è ciò che si fa nella maggior parte dei sistemi a nastro. In pratica, invece di spostare un file, si usa il resto dello spazio disponibile sulla cassetta, poiché le cassette forniscono una grossa quantità di memoria.

L'altro svantaggio è un problema di *accesso*: l'accesso è essenzialmente sequenziale. È difficile recuperare informazioni ad intervalli random (casuali).

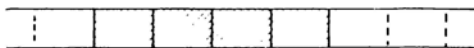



Fig. D-1: Un file sequenziale.

Records e blocchi

Al fine di facilitare l'allocazione della memoria ad un file, la maggior parte dei "sistemi di file" strutturano la memoria disponibile in blocks (blocchi). Questi blocks possono essere di dimensioni uguali, o di dimensioni variabili. Nella maggior parte dei casi, essi sono di dimensioni uguali. Essi saranno chiamati "settori", "pagine", "records", a seconda del sistema.

Ora un file userà vari blocchi, uno dei quali, l'ultimo, sarà di solito vuoto.

Così, in media, viene sprecato solo mezzo block di memoria. Come cresce il file? Abbastanza semplicemente, vengono assegnati al file, se disponibili, uno o più blocks.

Il problema successivo è quello di non perdere le tracce dei blocks assegnati al file, poiché è probabile che essi non siano in sequenza.

Strutturazione di un file

Possono essere usate due strutture base: un *directory*, o un *linked list.*

Un directory (elenco nominativo o direttorio).

In una struttura a directory, un "directory block" (o un directory file, che usi vari block) contiene tutti i "pointers" (puntatori) di tutti i blocks che appartengono al file

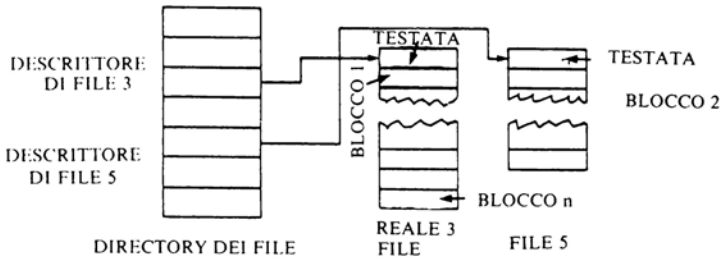



Fig. D-2.

Per esempio, su un diskette, il directory per il file "PAYROLL" contiene

1 - 2
1 - 3
1 - 4
1 - 5

Questo può essere letto come

il primo block è il settore 2 della traccia 1
il block seguente è il settore 3 della traccia 1
il block seguente è il settore 4 della traccia 1
il block seguente è il settore 5 della traccia 1

Questo è un modo semplice e diretto. Tuttavia per essere efficiente esso richiede che il directory risieda e sia letto nella memoria centrale del computer. Alla fine di ogni block, bisogna accedere al directory per determinare qual'è il prossimo.

Esercizio: *come può il sistema di file mantenere traccia dei blocchi disponibili, così da poterne assegnare uno o liberarne uno (o più)?*

Risposta: *esamina la fig. D-3 per la risposta.*

	0	1	2	3	4	5	6	7
0	0	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	0
2	0	0	0	0	1	0	0	0

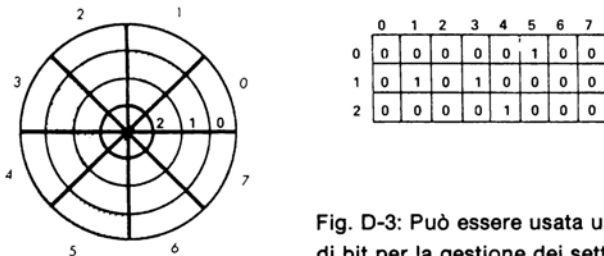



Fig. D-3: Può essere usata una mappa di bit per la gestione dei settori del disk.

Linked list (lista collegata)

Un'alternativa è quella di immagazzinare informazioni in ogni block. Questo è illustrato in fig. D-4. Ogni block contiene un "pointer" per il successivo block appartenente al file. Un segnale speciale, "EOF" (End of File-fine del file) indica la fine della linked list.

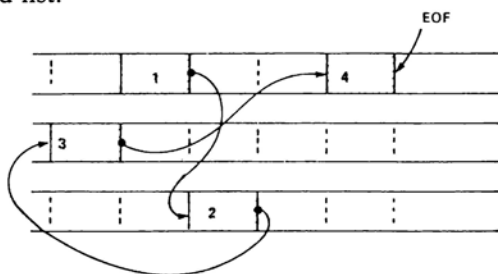



Fig. D-4: Linked blocks.

Indexed sequential (sequenziale a indice)

Possono essere usati metodi ibridi, come un directory (un "indice") di file sequenziali. Questo viene chiamato accesso indexed sequential (sequenziale a indice), un miglioramento notevole rispetto al semplice sequenziale.

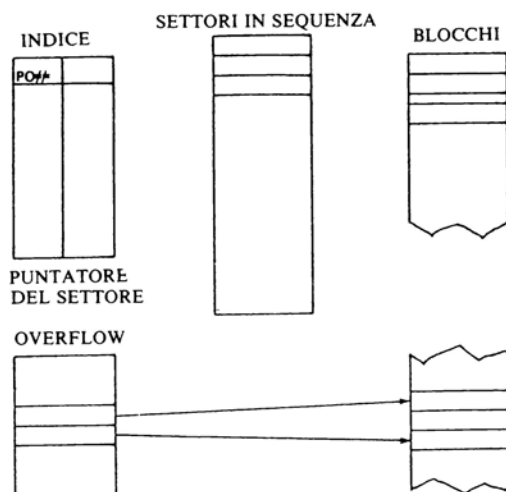



Fig. D-5: Immagazzinamento di ordini d'acquisto usando un recupero sequenziale ad indice.

RECUPERO DELLE INFORMAZIONI

Il prossimo problema da risolvere è quello di accedere efficientemente alle informazioni. In un file sequenziale, è necessario leggere un file intero per poter accedere a qualsiasi suo elemento. Questo è lento.

Il problema viene risolto strutturando. Più deve essere efficiente il meccanismo di accesso, e più strutturazione è richiesta. Strutturazione di solito significa dei directory in successione, o concatenazione complesse e comporta un tempo di accesso *minimo* più lungo. Riduce il tempo di accesso *medio*.

Trees

Uno dei modi più semplici per strutturare dei file è quello di usare una struttura ad *albero* (tree).

Un directory principale contiene i pointer per subdirectories (sub indici), e così via fino al file reale.

Per recuperare un file, si deve "camminare attraverso l'albero".

Possono essere usate altre strutture per immagazzinare informazioni e manipolare blocks all'interno del file stesso. Essi esulano dallo scopo di questo libro.

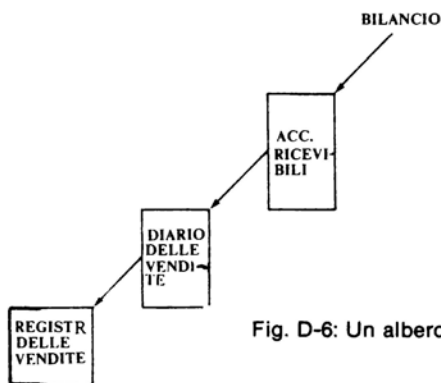



Fig. D-6: Un albero di liste commeciali.

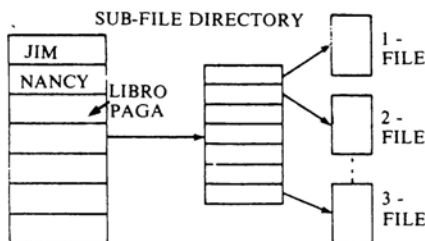



Fig. D-7: Un albero directory a 2 livelli.

APPENDICE E

ALCUNI COSTRUTTORI DI PICCOLI SISTEMI COMMERCIALI

Basic Four
14101 Myfard Rd.
Tustin, CA 92680

Burroughs
Burroughs Pl.
Detroit, MI 48232

Data General
Route 9
Southboro, MA 01772

Datapoint
9725 Datapoint Dr.
San Antonio, TX 78284

Digital Equipment Corporation
Parker St.
Maynard, MA 01754

Four Phase Systems
19333 Vallco Parkway
Cupertino, CA 95015

General Automation
1055 S. East St.
Anaheim, CA 92805

Hewlett Packard
11000 Wolfe Rd.
Cupertino, CA 95014

Honeywell
200 Smith St.
Waltham, MA 02154

IBM
Atlanta, CA 30301

Logical Machine
887 A Milten Rd.
Burlingame, CA 94010

Microdata
17481 Red Hill Ave.
Irvine, CA 92705

NCR
Main and K St.
Dayton, OH 45409

Olivetti
500 Park Ave.
New York, NY 10022

Philips
175 Froelich Farm. Blvd.
Woodbury, NY 11797

Quantel
3525 Breakwater Ave.
Hayward, CA 94545

Sycor
100 Phoenix Dr.
Ann Arbor, MI 48104

Wang
836 North St.
Tewksbury, MA 08176

APPENDICE F

COSTRUTTORI DI MICROCOMPUTERS

Alpha micro Systems
17875 N. Sky Park
North Irvine, CA 92714
(714) 957-1404

Apple Computer, Inc.
20863 Stevens Creek Blvd.
Cupertino, CA 95014

Commodore
901 California Ave.
Palo Alto, CA 94304

Cromemco
2400 Charleston Rd
Mountain View, CA 94043
(415) 964-7400

Data General
Southboro, MA 01722
(617) 485-9100
Telex: 48460

Digital Group
P.O. Box 6528
Denver, CO. 80206
(303) 777-7133

E & L. Instruments, Inc
61 First St.
Derby CT 06418
(203) 735-8774

Equinox Division
Parasitic Engineering
P.O. Box 6314
Albany, CA 94706

Extensys Corp.
592 Weddell Dr. S-3
Sunnyvale, CA 94086
(409) 734-1525

Heath Co.
Benton Harbor, MI 49022

IMS Associates, Inc.
(IMSAI)
14860 Wicks Blvd.
San Leandro, CA 94577
(415) 483-2093

MITS (Altair)
2450 Alamo S.E.
Albuquerque, NM 87106

North Star
2547 Ninth St.
Berkeley, CA 94710
(415) 549-0858

Polymorphic Systems
737 S. Kellogg
Galeta, CA 94608

Processor Technology
Box G, 7100 Johnson Industrial
Drive
Pleasanton, Ca 94566
(415) 829-2600

SWTC
219 W. Rhapsody
San Antonio TX78216

Technical Design Labs
Research Park, Bldg. H
1101 State Road
Princeton, N.J. 08540
(609) 921-0321

MICROPROCESSORI

Rodnay Zaks
Edizione Italiana: Jackson Editrice

Il testo base sui microprocessori per chiunque abbia una base tecnica e scientifica. Questo è un libro educativo, inteso ad insegnare passo dopo passo tutti i principi fondamentali dei microprocessori. Esso è il risultato dell'esperienza dell'autore nell'insegnare la progettazione del microprocessore a più di 2.000 persone. È stato definito praticamente da ogni lettore come "facile da leggere", "veramente istruttivo". Viene usato in tutto il mondo nelle principali università.

Questo libro tratta tutti gli aspetti dei microprocessori, dai concetti base alle tecniche avanzate di interfaccia, in presentazione progressiva. È indipendente da qualsiasi costruttore, e presenta principi e tecniche di progettazione uniformi e "standard", compresa l'interconnessione di un sistema "standard", come pure i componenti specifici. Esso introduce l'unità a microprocessore, come funziona internamente, i componenti del sistema (ROM, RAM, UART, PIO, altri), l'interconnessione del sistema, le applicazioni, la programmazione, ed i problemi e le tecniche per lo sviluppo di un sistema.

LESSICO DEI MICROPROCESSORI

Edizione Italiana: Jackson Editrice

Un libro di riferimento formato tascabile con tutte le definizioni del gergo del microprocessore. Da concetti elementari ("latch", "memoria") a componenti avanzati (floppy-disk-controller). Questo libro tascabile è completamente aggiornato e facilita la lettura di qualsiasi scritto riguardante i microprocessori o i microcomputer. Esso è pure uno strumento educativo che fornisce un accesso random (casuale) esauriente alle informazioni necessarie.
CONTENUTO: Dizionario (definizioni, acronimi), il gioco dei numeri, Segnali: S-100, RS-232C IEEE 488, Indirizzi di Costruttori, JETDS, Conversioni di Codice.

TECNICHE DI INTERFACCIAMENTO DEI MICROPROCESSORI

Austin Lesea e Rodnay Zaks
Edizione Italiana: Jackson Editrice

Interfacciare il microprocessore non è più un'arte. È un insieme di tecniche, e in alcuni casi solamente un insieme di componenti. Questo libro esauriente introduce i concetti base e le tecniche di interfaccia, quindi presenta con precisione i dettagli di realizzazione, dall'hardware al software. Esso tratta tutte le periferiche essenziali dalla tastiera al floppy disk, come pure i bus standard (dall'S-100, all'IEEE 488) e introduce le tecniche base per l'individuazione dei guasti.

CONTENUTO:

1 - INTRODUZIONE: concetti, tecniche base, segnali di controllo del microprocessore.

2 - ASSEMBLARE LA CENTRAL PROCESSING UNIT: Introduzione, architettura del sistema, addressing. Il sistema 8080, il sistema 6800, lo Z-80. Interfaccia per RAM dinamica. L'8085.

3 - INPUT-OUTPUT BASE: Memoria contro I/O mapping. Input-Output parallelo: tecniche e chip (PIO). Input-Output seriale: programma ed UART. I tre metodi di controllo input-output: polling, interrupt, DMA. Circuiti utili.

4 - INTERFACCIARE LE PERIFERICHE: Tastiera, bounce, codificare, rollover. Display LED Teletype. Carte - lettore di nastro. Line printer. Lettore di carta di credito a banda magnetica. Interfaccia per cassette. Display CRT. CRTC. Floppy disk. FDC. CRC.

5 - CONVERSIONE ANALOGICO - DIGITALE: D/A concettuale, D/A pratico, prodotti reali, l'A/D, teorema del campionamento, approssimazione consecutiva, integrazione, conversione a confronto diretto. ADC e DAC. Interfacciare i D/A e gli A/D. Sottosistema di raccolta dei dati, scoling, offset, conclusione.

6 - BUS STANDARD: Paralleli: S-100, 6800, IEEE - 488, CAMAC, Seriali: EIA - RS-232C, RS-422, RS-423, formati sincroni.

7 - STUDIO DI UN CASO: UN MULTIPLEXER A 32 CANALI: Introduzione, specificazioni, architettura, software, modulo CPU, modulo RAM, modulo USART, modulo di interfaccia dell'Hosk computer, conclusione.

8 - INDIVIDUAZIONE DIGITALE DEI GUASTI: I problemi. Cosa c'è che non va: componenti, rumore, software, gli strumenti ed i metodi: VOM, DVM, oscilloscopio, prove logiche, signature analysis, emulazione, simulazione, analizzatori di stati logici. Studio dei casi: storia dei guai. Il banco perfetto.

9 - CONCLUSIONE - EVOLUZIONE: I nuovi chip, sistemi a chip singolo. Software plastico.

INTRODUZIONE AL PERSONAL E BUSINESS COMPUTING

di Rodnay Zaks

Un'introduzione esauriente, tuttavia semplice al mondo dei microcomputer per l'utente potenziale. Un'introduzione dettagliata ai concetti, alle periferiche, alle tecniche. Questo libro è stato concepito come testo didattico e serve sia per i principianti che per gli utenti che cercano di ampliare la loro conoscenza e preparazione.

NOTE SULL'AUTORE

Rodnay Zaks si è laureato in fisica all'Università della California Berkeley. Egli si è interessato nella reale applicazione industriale dei microprocessori sin dal loro inizio. Ha tenuto seminari e lezioni sull'uso e la tecnologia dei microprocessori a più di 3000 persone in tutto il mondo. È l'autore di più di 30 pubblicazioni o scritti di ricerca. Questo libro, come altri della serie, è il risultato diretto della sua esperienza tecnica e didattica.

ALTRI LIBRI IN QUESTA SERIE

MICROPROCESSORI: dai chip ai sistemi
TECNICHE D'INTERFACCIAMENTO DEI MICROPROCESSORI

L. 14.000
(13.208)

JACKSON
ITALIANA
EDITRICE

RODNAY
ZAKS

INTRODUZIONE AL PERSONAL
E BUSINESS COMPUTING

18